COMPUTATIONAL SIMULATION OF ADAPTATION OF WORK STRATEGIES IN HUMAN-ROBOT TEAMS

A Dissertation
Presented to
The Academic Faculty

by

Martijn IJtsma

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2019

COMPUTATIONAL SIMULATION OF ADAPTATION OF WORK STRATEGIES IN HUMAN-ROBOT TEAMS

Approved by:

Dr. Amy. Pritchett, Advisor
Department of Aerospace Engineering
Pennsylvania State University

Dr. Glenn Lightsey
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Karen Feigh, Co-Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Matthew Johnson
Institute for Human and Machine Cognition

Dr. John-Paul Clarke
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: July 3, 2019

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor Dr. Amy Pritchett. Her enthusiastic encouragement, thought-provoking questions, and patient feedback have been paramount to this research work and my development as a researcher. This gratitude extends to my co-advisor Dr. Karen Feigh, whose sharp insights and experience have been very valuable in the development of this work. I am also very grateful to the other committee members, for taking considerable time and effort to review and help improve this work.

I am grateful to the staff of the School of Aerospace Engineering, who have made me feel welcome here, and to NASA for funding part of this work. I would also like to thank my professors at Delft – Dr. Clark Borst, Dr. René van Paassen and Dr. Max Mulder – for the encouragement and advice they provided during the first half of my graduate career, and for inspiring me to continue in research.

I am very grateful to the other students that worked on WMC, before me and with me, with a special mention to Raunak Bhattacharrya, Lanssie Ma, Abhinay Paladugu, and Sean Ye. My gratitude is also extended to my other lab mates with whom I have shared the ups and downs of graduate school: Andrew Greenhill, Lee Witcher, Will Lassiter, Melissa Baltrusaitis and other CEC'ers. A special mention to Michael Portman for the countless writing sessions we held together.

Finally, I am extremely grateful to my wife, Celine Admiraal, who came with me on this amazing adventure and has been ever supportive of me. I would also like to thank my

parents and sister, who had to do without me for the past years, but were always encouraging and caring.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ADS	Abstraction-Decomposition Space
CRM	Crew Resource Management
CSE	Cognitive Systems Engineering
CWA	Cognitive Work Analysis
EV	Extra-vehicular astronaut
EVA	Extra-vehicular activity
HCI	Human-Computer Interaction
HTN	Hierarchical Task Network
IR	Information resource
ISS	International Space Station
IV	Intra-vehicular astronaut
MCC	Mission Control Center
OPD	Observability, Predictability and Directability
PR	Physical resource

QOS	Quality-of-Service
RMS	Remote Manipulator System
SRK	Skill, Rule, Knowledge
WDA	Work Domain Analysis
WMC	Work Models that Compute

SUMMARY

Resilience describes a socio-technical system's ability to adapt to uncertainty, both accounted-for and unanticipated in the design process (Woods, 2002). Socio-technical systems operating in complex work domains, such as space operations, need to be resilient to maintain performance under a wide variety of work conditions. When designing socio-technical systems, one way of fostering resilience is to build support for humans in the system to adapt fluently to changing demands.

Design approaches accounting for the need for flexibility and adaptation in teams operating in complex work domains emphasize the need for formative insight that help understand how a team's collective is driven by characteristics of the work itself, the work environment and the agents. Several models have expanded on these ideas, but further systematicity is desired for providing formative and comprehensive insight into the effects of work allocation and interaction modes on team adaptation. In addition, current methods are typically static in nature, not explicitly accounting for the temporal dynamics of a team's coordination of activities. This thesis proposes an approach based on computational modeling of a team's collective work to analyze adaptation of work strategies in teams, specifically how work allocation and interaction modes can foster and/or limit such adaptation.

The modeling of adaptation is performed from a work-perspective, identifying where characteristics of the work and interaction within a team limit available work strategies and the associated effects on adaptation. With computational models of work presented in this thesis, the dynamic and emergent nature of the problem can be analyzed directly, providing

insight into how the team's interweaving and coordination of dependent work affects the feasibility and appropriateness of multiple work strategies. Moreover, the computational modeling of work provides a powerful basis for design in that, through quick "what-if" experiments, the design trade-space for teams can be mapped with minimal modeling and experimentation effort and time.

This thesis also explicitly addresses the challenge of design human-robot teams, wherein technological agents work alongside human agents as team mates. A challenge when introducing technological agents into a team (e.g., automation or robots), the rigidity of technology often limits a team's ability to adapt. The computational modeling of work can help predict the effects of such limitations on the team's effectiveness in terms of the feasibility of different work strategies, by directly evaluating the temporal dynamics of how human and robot need to coordinate their collective work.

The approach presented in this thesis is demonstrated in two case studies, in the work domain of manned spaceflight operations. Space operations is characterized by high uncertainty in the work domain, sparsity of manpower and resources, and lack of accurate testing environments. Additionally, for future missions, significant communication delays will require many of the tasks currently executed by the mission control center to be performed by the space crew, with the support of robotic team members. The computational models of work provide a systematic and formative analysis method to support mission planning.

CHAPTER 1. INTRODUCTION

For achieving resilience, a team of workers must be able to adapt fluently to new work demands. One of the unmatched capabilities of humans, compared to robots and automation, is that humans are able to fluently adapt to new, unforeseen circumstances in the work environment, changing their behavior to meet work demands while keeping their workload and performance at acceptable levels. In practice such adaptation occurs naturally in human teams, as team members adapt their behavior and fluently synchronize their activities with those of others. This adaptation allows a system to deal with uncertainty in the work environment and the work itself.

When new automation or robotic support aids are introduced into the operation, however, it is frequently observed that the rigidity of the technological aids limits the human's, and therefore the team's, ability to adapt. One cause of this rigidity is that in the design of these technological aids, assumptions are made about how the team ought to execute its work (work-as-imagined). As these assumptions are then reflected in the design of the team's operations, the technological aids impose a particular allocation and strategy of work on anybody who is working with the support system (Woods and Hollnagel, 2006). However, when work-as-imagined assumes a work strategy not appropriate for all situations, the introduction of these support systems makes the team's operations unnecessarily rigid, rather than allowing for different strategies as a form of adaptation. This operational rigidity frequently results in the disuse of automation or robotic support in the face of unanticipated work demands, or worse, in a brittle system that fails to perform under certain work conditions.

The adaptation of human workers to changing work demands has been studied extensively (Entin and Serfaty, 1999; Gauthereau and Hollnagel, 2005; Hollnagel, 1993). These studies have focused primarily on the modeling of work strategies and adaptation of individuals, see for example Hollnagel's work on contextual control (Hollnagel, 1993). This work highlighted that human workers adapt their strategies (a process referred to as 'control') in response to contextual factors, most notably the perceived available time to complete tasks. In this and most other existing models, an individual's adaptation is modeled as reasoning about feasible sequences of actions, each with relative benefits and limitations. For example, some sequences may have higher performance, but at the cost of requiring more information and knowledge, and of being more effortful. From a work-perspective, adaptation is then constrained and driven by dependencies in the work, such as availability of resources or ordinality relationships between actions.

Related research has looked at human strategies for using automation aids, and why and in what situations certain automated aids go unused (Kirlik, 1993). This work highlighted the effects of required interaction overhead for off-loading tasks to an automation aid: Whether or not a work strategy involving the support aid is beneficial to the team depends on how much time and effort it takes to interact with the aid, as well as other contextual factors such as secondary tasks that need to be completed. Thus, the types and characteristics of interaction that an aid requires determines the strategies that are available, and how attractive each of these strategies is to the human operator.

Insights from this earlier work relate mainly to individual adaptation, or to human-automation pairs. Little research on adaptation has explicitly addressed work strategies in teams involving multiple human workers, possibly working with automation or robotic

aids. The research efforts discussed in this thesis have the aim to analyze adaptation in teams, specifically how allocation of work and interaction between team members can foster and/or limit such adaptation.

1.1 Research Questions, Objectives and Scope

This thesis views effective team design as supporting flexibility in strategies for conducting the team's collective work, spanning a wide range of potential operating conditions. In addition, effective teaming requires strategies to support the dynamics of the work, with the work of team members fluently coordinated and interweaved. With this definition of effective team design, the high-level design question that is the inspiration for this thesis can be formulated as:

> How can team design support flexibility in strategies for conducting a human-robot team's collective work?

To gain insight into this high-level research question, this thesis takes a work perspective, as opposed to human- or technology-centered perspectives. It is hypothesized that a systematic analysis of the constraints and characteristics of the team's work and work domain can inform an answer to this question and help guide the conceptual design of human-robot teams. For effective human-robot team design as described at the beginning of this subsection, the constraints and characteristics of interest particularly relate to the team's use of tools, sharing of information and interaction between team members, which, from first principles, affect the team's collective feasible work strategies and the required coordination. A challenge is that the complexity of these work domain constraints and the interactions thereof often manifest in emergent ways, complicating the prediction of how

design decisions will affect a team's behavior. In that light, this thesis addresses two research questions in specific:

- How do constraints in the work limit and/or drive a team's work strategies, particularly looking at behaviors that extend to the use of tools, sharing of information, and interaction between team members?
- What aspects of human-robot team design foster appropriate selection of work strategies and effective adaptation?

An understanding of these questions can help identify potential issues related to adaptation early in design, before significant investments and commitments are made towards particular technological solutions or prototypes. Thus, the modeling in this thesis aims to create a systematic basis for inquiry into potential design issues, to provide formative insight on the part of the designer, as opposed to directing the design to a single optimal solution for conducting the team's work. Moreover, with its focus on conceptual team design through work analysis, this thesis will not explicitly consider, or model, internal cognitive processes associated with adaptation, nor will it analyze specific robotic algorithms or technology for (real-time) planning of operations. Finally, the thesis considers teams that consist of anywhere between two to ten agents, which are representative team sizes for many of the complex work domain that this work would be relevant to (e.g., manned spaceflight missions have teams usually consisting of three astronauts and the mission control center).

1.2 Document Structure

Chapter 2 is the literature survey that explores concepts of effective teaming, methodologies for team design, and challenges and methods for human-automation and human-robot teams. Chapter 3 then introduces work modeling for teams and the concept of work dynamics, which is central throughout this thesis. Chapter 4 addresses identification of feasible work strategies, based on the inherent characteristics of the work and the work domain that the team operates within. Chapter 5 discusses team design in terms of work allocation and definitions of interaction modes, relating work dynamics and work strategies to considerations of coordination and teamwork in multi-agent teams. Chapter 6 focuses on the analysis of works strategies and work allocation in human-robot teams specifically, describing how common issues with human-robot team can be analyzed using concepts introduced in earlier chapters. From this, several implications are derived for human-robot team design and a human-robot team's ability to adapt. Chapter 7 describes conclusions and contributions of this work, and considerations for future work.

Throughout chapters 3-6, two case studies are discussed to illustrate the concepts introduced in each chapter. These case studies are meant as demonstration and will hopefully assist the reader in clarifying how the approach laid out in this thesis can lead to useful insights. The work domain of interest for these case studies is manned spaceflight operations. NASA and other space agencies envision, in the foreseeable future, an increased use of robotic and automation aids in manned spaceflight operations. New robotic technologies should assist astronauts and off-load part of their heavy taskload. As these systems are introduced into space operations, which are characterized by the unpredictability of the space environment, it is imperative that the astronaut's ability to adapt to new work demands remains intact.

CHAPTER 2. LITERATURE SURVEY

This chapter presents a literature survey on the state-of-the-art of the designing of teams. First, section 2.1 examines the scientific consensus on effective teaming: what makes a team effective? Then, section 2.2 examines methods for analyzing and designing teams, specifically in terms of the models that are available, the concepts of teaming they generally focus on, and how they relate to other perspectives on socio-technical systems. In section 2.3, specific challenges associated with human-robot teams are discussed. Section 2.4 discusses the work domain of manned spaceflight operations, which in this dissertation is used as an example domain. Finally, section 2.5 relates the contributions of this dissertation to the current state-of-the-art are.

2.1 Teams in Complex Work Domains

2.1.1 Definitions

This dissertation is on the design of teams working in complex work domains. Woods (1988) and Rasmussen and Vicente (1990) describe several characteristics of these domains, including the dynamic and event-driven nature of their environments: they often include dynamic processes that need to be controlled or accounted for by human workers, such as aircraft dynamics in air traffic control (Ahlstrom, 2005), or nuclear reactors in monitoring of power plant operations (Moray, Sanderson, and Vicente, 1992). Second, complex work domains involve many interconnected and interacting parts or subsystems. Third, there is considerable risk involved. Finally, there is uncertainty in the data available to human workers in the system. In this dissertation, the focus is on systems with the first

three characteristics, i.e., dynamism, components that are highly connected and safety-critical.

Also, characteristic to complex work domains is the high variability in work demands that workers need to deal with. Variability stems from both the work environment (e.g., environmental disruptions such as weather) and the work system, which include the agents and the artifacts through which work is performed (e.g., changes in consumable levels). Some of these variations are foreseeable at different time scales; others are unforeseeable.

A team is defined as "a number of persons associated together in work or activity" (Merriam-Webster). A team is involved in joint activity defined as "an extended set of actions that are carried out by an ensemble of people who are coordinating with each other." (G. Klein, Woods, Bradshaw, Hoffman, and Feltovich, 2004). Through joint activity, the team members are interdependent and working towards the same goal.

Recent perspectives on the design of socio-technical systems has strived for a perspective of technology as a "team player" (G. Klein et al., 2004). In such a perspective, instead of technology as a tool or prosthesis, it is viewed as possessing "agency", either amorphous (e.g., software agents) or embodied (robotic agents), which is discussed in more detail in section 2.3. Thus, as opposed to "persons" that comprise a team, the term "agents" will be used to account for both humans and technology that are considered to have agency.

2.1.2 Variability and Adaptation in Complex Work Domains

From a highest-level perspective, an effective team attains the work goals it was built for in an efficient and safe manner. Efficiency can be defined as the state of "producing desired

results with little or no waste (as of time or materials)" (Meriam-Webster). Thus, an efficient team minimizes the time and/or material usage, or resources consumption, throughout its operation. Safety is a more elusive objective and can be defined in multiple ways and from different perspectives. In its simplest definition, safety is "the condition of being safe from undergoing or causing hurt, injury, or loss" (Merriam-Webster). Often referred to as Safety-I, this definition corresponds to a view where safety is the absence of any harm (Hollnagel, Wears, and Braithwaite, 2015).

In the face of new work demands, whether foreseen or unforeseen by the original designers of a work system or by the human workers themselves, humans are able to strategize their work such that resources and demands are held in continuous equilibrium without sacrificing performance (Hollnagel and Bye, 2000). For example, Sperandio (1978) showed that air traffic controllers gradually change to more economical (i.e., less effort-full) strategies as work demands increase and manage to maintain performance without violating critical workload levels. Feigh and Pritchett (2005) studied airline operations management and observed how sector managers employ different strategies, ranging from elaborate formulation and consideration of multiple solutions, to quick decisions with minimal information retrieval, depending on how demanding the work is and how much time is available.

More advanced definitions of safety describe safety not as a binary condition that can be attained through design by diminishing any potential for failure. Safety-II corresponds to a view where "failures" and "successes" occur for the same basic reasons and was proposed as an "alternative or complement to conventional views of safety" (Hollnagel et al., 2015). In line with the ability of human workers to adapt to changing work demands,

the safety-II perspective defines safety as a continuous process of adaptation through human activity. Things go right not because the human workers abide by the procedures, but because the system adapts to meet contextual work demands (Hollnagel et al., 2015). It is because of this adaptation that the system operates successfully. Failures, then, are the consequence of the system not adapting to variability in the work demands.

How well a system adapts to changing work demands is described as the system's resilience. For resilient performance, a system needs to be able to respond to disturbances, to monitor for events that could affect its performance, to learn from prior experiences and to anticipate future developments (Hollnagel, 2011). Woods (2002) notes that, beyond just adaptation, resilience is concerned with the deeper ability of a system to "handle disruptions and variations that fall outside of the base mechanisms/model for being adaptive as defined in that system". Thus, resilience, first, requires a thorough understanding of how the system adapts to new demands, but then, second, an understanding of how boundary conditions may challenge this adaptation. In light of this second aspect of resilience, Woods introduces the concept of "graceful extensibility", which refers to the notion that a resilient socio-technical system does not fail abruptly but instead degrades or even extends performance beyond its design envelope (Woods, 2015, 2018).

Thus, an effective team continuously adapts its work to respond to changing work demands. For a team to effectively maintain safety and efficiency, the team as a whole must first be aware of how safety and efficiency could be affected throughout its operation. Second, it needs to be able to control the system, including itself, to prevent loss of safety and/or efficiency.

2.1.3 Control and Coordination in a Team

Expert human workers change their behavior in response to work demands in the form of contextual factors such as subjective available time (Hollnagel, 1993). Hollnagel created a model of different types of human behavior, each appropriate for different contexts. This model was a response to, at the time, ubiquitous linear models of work that describe the human worker going through an "optimal" sequence of actions. However, in actual operations, time constraints and limited resources often demand other, more cognitively efficient types of behavior. Thus, Hollnagel decoupled the model of competence, describing all actions a human workers can take, from the model of control, describing how specific actions are selected and sequenced by the worker him/herself.

The model of control consists of different control modes, each with different behavioral aspects, and each appropriate in a different context. The control modes range from scrambled control (or panic mode) to strategic control (or optimizing mode), and vary in the degree of deliberation that a human worker puts in deciding what actions to take next. For example, in so-called opportunistic control, only one action at a time is selected, in response to a single environmental stimulus.

It is an open question as to what degree one can classify an entire team's behavior according to the contextual control modes, or whether each team member individually is in a control mode, where team members' modes can overlap but could also be different. Stanton, Ashleigh, Roberts, and Xu (2001) provide some indicative evidence that it is possible to classify an entire team's behavior according to Hollnagel's four control modes.

Other indicative evidence include studies how teams switch from explicit to implicit coordination strategies under stricter time constraints (Entin and Serfaty, 1999).

Within a team, the control of action sequences is dependent on what other team members are doing. Thus, some control must be coordinated within the team. Coordination in a general sense is defined as "the effective alignment and adjustment of the partners' actions" (Merriam-Webster). In the context of a team, Gary Klein, Feltovich, Bradshaw, and Woods (2004) define coordination as "the attempt by multiple entities to act in concert in order to achieve a common goal by carrying out a script they all understand." In a related publication (G. Klein et al., 2004), coordination is described in close relation to joint activity in the context of "how activities interweave and interact".

Thus, through coordinating their activities, effective teams interweave the work to make efficient use of time and resources. For time efficiency, the work is distributed in the team such that all team members can work continuously and on tasks that are directly necessary for achieving the work goals. Ineffective coordination would be a situation in which team members need to wait unnecessarily, thereby wasting time. For efficiency in resource utilization/consumption, effective coordination can make the best use of consumable resources. Likewise, artifacts that are used to conduct work but must be shared between team members need to be managed smartly. This means that these resources must be managed spatially as team members work in different locations and must be handed-over from one agent to another.

Several studies of effective teaming have identified that, for effective coordination of joint activity, agents must be observable, predictable and directable (Christoffersen and

Woods, 2002; Gary Klein et al., 2004). Coordination requirements can therefore be described in these three categories: the observability, predictability and directability (OPD) requirements (Johnson et al., 2014; Woods and Hollnagel, 2006). Observability relates to the ability of agents to observe the state of other agents in the team and is required for team members to maintain awareness of what is going on within the team. Such awareness is needed for synchronizing activities within the team and additionally, especially in safety-critical domains, serves to verify and check that no mistakes are being made or anomalies are missed.

Predictability denotes that each agent's actions should be "predictable enough that others can reasonably rely on them when considering their own actions." (Johnson et al., 2014). When considering one's own work strategies, one needs to make assumptions about what other agents are reasonably going to do in the future. Predictability in other agents facilitates synchronization of dependent work and thereby the selection of appropriate work strategies. Thus, with predictable agents, work strategies for teams can be constructed such that they maximally make use of opportunities for synchronizing actions by pro-actively managing work dependencies.

Finally, directability relates to the ability of agents to direct the actions of other agents. This is especially relevant in hierarchical frameworks wherein one agent determines actions for other agents. When automated agents are involved, directability is desired for the human operator to be able to override the automation and provide necessary guidance.

Studies of coordination in teams have furthermore looked at the role of shared mental models, encompassing each team member's knowledge of the other team members'

capabilities and actions. A recurring finding in communication theory is that, in effective teams, team members provide information before it is requested by other team members. Such proactive information exchange allows the team to maintain performance during periods of high workload. For example, Entin and Serfaty (1999) noted that when team members are sensitive to each other's workload and state, they can anticipate each other's needs and communicate information pro-actively. Shared mental models of the team, the task and the information requirements help foster such proactive communication (Stout, Cannon-Bowers, Salas, and Milanovich, 1999).

Planning prior to starting operations can help build shared mental models (Stout et al., 1999). Likewise, procedures and coordination protocols can support managing OPD requirements in the face of changing work demands. For example, in the aviation domain, crew resource management (CRM) is used for maintaining awareness of each other's actions and managing errors, in which cross-checking and verification procedures between team members (human and automation) are used to catch any errors and/or mitigate their effects (Helmreich, Merritt, and Wilhelm, 1999). In adaptation of work strategies, such procedures and protocols can serve as templates that are readily available to the human operator (through training and experience, or through procedures that can be physically accessed during operations), thereby mitigating the mental efforts required of the adapting human team member.

2.1.4 Summary

Teams working in complex work domains need to be able to adapt to match changing work demands with available resources. Human workers are exceptionally good at adaptation,

being able to change their behavior to maintain performance and acceptable workload levels under a variety of work conditions. In a team, however, this type of control must be coordinated with other team members. Effective teams interweave their activities in such a way that time and resources are effectively used. To coordinate activities, other team members must be observable, predictable, and directable. Shared mental models, planning and procedures can help streamline coordination.

2.2 Building/Designing Teams

The previous section explored what characteristics make an effective team; this section discusses the state-of-the-art on how to design teams with these characteristics. Design of a team could be defined as determining the operational framework in which the individual workers will operate. It encompasses design decisions such as determination of the team composition (e.g., how many team members, what technological agents to employ?), their roles and responsibilities (e.g., is there a single supervisor, what is the role of technology?), the team training (e.g., how are team members trained to communicate and coordinate?), and the communication guidelines (e.g., how will each team member communicate with the others?).

In this dissertation, the focus is on conceptual design phases. The design decisions made in these phases have a major impact on the team's operations, yet in these early stages it is difficult to predict or evaluate how different options will play out in actual operations, as very little information is available and opportunities for testing through human-in-the-loop experiment are limited. This dissertation furthermore focuses on three aspects of

conceptual design of team frameworks: determination of the team composition, work allocation, and determination of the interaction protocols.

Team composition is the determination of the number and type of agents in the team. Teams can vary in size, but the focus in this dissertation is on teams smaller than about ten members. Team members can be co-located or working remotely from each other. Different combinations of co-located and distributed team members are possible. Additionally, with technological agents as team members, there is also a question of how what the ratio between technological versus human agents should be (Defense Science Board, 2012).

It is assumed that the humans in a team are expert workers, who possess the necessary knowledge and skills to effectively control the system. In conceptual design phases, one would be less concerned with issues associated with individual differences between human team members, as well as any learning effects – instead, the aim is to support-through-design the expert team. This in line with much of the literature from cognitive engineering that was heavily cited in the previous section, which is specifically focusing on supporting expert workers in complex work domains.

Work allocation is the distribution of authority and responsibility in the team. Traditional methods focus just on the allocation of authority, denoting which agent is executing each part of the work. As argued by Feigh and Pritchett (2014), work allocation should also include considerations of responsibility, denoting which agent is accountable for the outcome of an action. Other terms for work allocation include function allocation (Feigh and Pritchett, 2014), and task allocation (Singer and Akin, 2009).

Interaction protocols define the types of teamwork for coordinating work between team members. As work is distributed in the team, additional teamwork is necessary to coordinate work (Feigh and Pritchett, 2014). What teamwork mechanisms are employed, however, are a design decision on its own. Examples of teamwork mechanisms include needing confirmation from other team member(s) before proceeding with follow-up actions, and team members providing instructions or commands to other team members.

Even with this scoping of the design problem, design of teams for complex work domains is a challenging undertaking. In particular, the complexity of the system (e.g., variability in work demands and the team's required adaptation to meet these work demands), as described in the previous section, make it impossible for designers to foresee all possible situations that a team will encounter during its operation. These characteristics make it challenging to optimize design in the traditional engineering approach.

2.2.1 Formative vs. Normative Design

An assumption that is often made in the design of socio-technical systems is that design should enable human operators to optimize for maximum performance in terms of a few, core system objectives. However, when considering different contexts in which human workers need to operate, it is necessary to recognize that optimizing one's work strategy for performance does not always match with the available resources. As described in the previous section, when the resources for meeting work demands are not sufficient, human workers maintain performance by switching to work strategies requiring fewer resources. Thus, to design a system that is resilient, this human ability to adapt work strategies to

context should be supported in design of teams, in the form of flexibility and viable work strategies that are suitable for different contexts and varying work demands.

Design should not be "normative" in the manner of many linear processing models used historically to design cognitive systems. Normative modeling frameworks prescribe an "optimal" work strategy that, in practice, limits adaptation. Instead, design should purposely support flexibility within a socio-technical system to allow human workers to adapt when faced with unanticipated situations (Vicente, 1999).Thus, methods should instead be "formative," providing the designers with ways to inform design, but then allow the human (expert) workers to "finish the design" as appropriate in the context.

Cognitive Work Analysis (CWA), a design framework central to CSE, advocates for a formative approach that puts affordances and constraints of the work domain and the work as central constructs in the analysis and design of sociotechnical systems (Vicente, 1999). To support flexibility, the modeling is focused on capturing ecological and cognitive constraints that remain constant regardless of the specific context but shape the resulting behavior of the system. By basing the design of socio-technical systems on constraints that are inherent to the work environment and the agents within the team, designers can avoid imposing a particular way of doing things on the human workers, and enable the workers to employ and adapt their own work strategies.

CWA also formalizes the idea that an effective socio-technical system allows human workers to adapt to work demands through the support for different strategies (Vicente, 1999). Vicente defined strategies as the processes by which control tasks can be achieved. Whereas the definition of the task itself is a simple description of input and output,

strategies define the possible ways by which a task's output can be realized. In earlier but similar work, Rasmussen (1981) formally defined a strategy as "a category of cognitive task procedures that transforms an initial state of knowledge into a final state of knowledge." Rasmussen purposely defined strategy as a category, to contrast it with individual task procedures which are isolated, non-recurring instances. Context requires slightly different procedures in each situation, but, as Rasmussen argued, the specific instances of these procedures can be categorized based on similar and stable characteristics.

Earlier work has applied CWA methods to study work in teams (Ashoori and Burns, 2013; Miller, McGuire, and Feigh, 2017). However, the formative methods proposed in CWA have been based on qualitative methods that, relative to the needs of designers not versed in CWA, can be rather vague and do not model the temporal dynamics of the team's joint work. The methods additionally are manual and therefore labor and time intensive (Bodin and Krupenia, 2016). Possibly due to these reasons, the design framework and its philosophy have not gained widespread adoption beyond the cognitive systems engineering community. A more systematic approach to formative design is therefore needed.

2.2.2 Static vs. Dynamic Analysis for Design

A second distinction that can be made in design methodologies is between static and dynamic analysis of the system-to-be-designed. Most methods in cognitive systems engineering qualify as static methods. The high complexity of the problem have led to a series of qualitative models that allow designers to describe and analyze a system (Ashoori and Burns, 2013; Johnson, 2014; Vicente, 1999). However, there is no explicit, quantitative analysis of the time responses of the system, in contrast to traditional engineering

approaches for technology design such as control systems design. Any dynamic analysis is usually performed through field observations or human-in-the-loop studies, from which static, qualitative models are deduced. However, there is a small collection of methods that do account for dynamic effects, which will be discussed in the following paragraphs.

A commonly used simulation frameworks for human performance modeling and system integration is the Improved Performance Research Integration Tool (IMPRINT), developed by the U.S. Army Research Laboratory (Army Research Labratory, 2009; Laughery, 1999). IMPRINT is used – mainly in military applications – to determine the number of personnel required for a mission and the task allocation, and can evaluate mission designs for issues related to performance and workload. It consists of a discrete-event simulation of functions and tasks that are linked through paths, which is defined beforehand by the modeler as a "task network" (Laughery, 1999). During a simulation, "entities" move through the task network and are manipulated by the task models, while human performance models estimate workload. Thus, although a mature simulation framework with a wide variety of applications (Boeke, Miller, Rusnock, and Borghetti, 2015; Colombi et al., 2012; Pomranky, 2006), the framework assumes normative work sequences through the definition of task networks, thereby limiting the consideration of adaptability and a range of different strategies for executing the work.

Studies in robotics have looked at metrics for fluency of collaboration between a human and a robotic agent that are based on an evaluation of the dynamic behavior of collaboration (Hoffman, 2013). Moreover, in planning and AI, there is a large body of research on Hierarchical Tasks Networks, from which a Temporal Plan Network can be created to compute action sequences and explicitly consider the temporal aspects of tasks (D. E.

Smith, Frank, and Jonsson, 2000). These analyses are used in real-time in AI and planning, but can also be used in design of human-robot teams (Shah, Saleh, and Hoffman, 2007). Although this appears to at least provide the option to do a broader analysis beyond just individual work flows as in IMPRINT, the HTN explicitly specifies the ordering constraints in terms of what actions should be used for particular purposes. Indeed, a critique of researchers is that HTN methods are "closer to "programming" a particular application, rather than providing a declarative description of available actions and using general techniques to do the planning." (D. E. Smith et al., 2000).

In the HCI domain, a good example of dynamic analysis through simulation is the Brahms framework, which has been used to study adjustable autonomy in human-autonomy teaming (Sierhuis et al., 2003). Brahms is an agent-based simulation tool that can be used for modeling "work practices" in situated activity. It was later integrated with KAoS, a modeling framework for teamwork, mobility and resource control. Both employ a policy method to model their respective parts. Together the models can create a full-fledged software agent development platform to study teamwork and collaboration between different entities of a system. Brahms has in the past been applied to study ISS operations (Cquisti, Sierhuis, Clancev, and Bradshaw, 2002), but requires existing data of activity schedules, including their ordering, as basis for simulation.

It should be noted that dynamic analysis is different from planning of a team's activities and scheduling of resources. The problem formulations typically rely on known team compositions, agent capabilities and task definitions. Moreover, the aim of the problem formulations in planning and scheduling is to find one optimal solution, commonly in terms of time-based objectives (e.g., minimizing makespan and/or idle time), not to provide

formative insight into how a team would need to adapt and coordinate work. For example, a common problem formulation in scheduling is the job shop (Błazewicz, Domschke, and Pesch, 1996) . Job shop scheduling is an optimization problem in which a given set of jobs, with varying durations, needs to be assigned to a given number and type of machines, with the objective of minimizing makespan. Solver methods can be applied to optimize a team's performance and can support a team's plan adaptation under uncertainty, prior or during a specific mission. For example, Adams et al. (2002) proposed a hierarchical architecture for planning and control for teams of unmanned aerial vehicles that specifically addressed a team's adaptation in dynamic environments. However, what is needed in conceptual design of teams is broader and formative insight into the dynamic and emergent behavior of a team, given different team designs. Methods for finding optimal solutions for allocating and planning work in teams (as a normative solution), whether prior to a mission or in real-time, do not provide this broader insight.

However, these domains have concepts that can be used to analyze the temporal behavior of a team, as these problems are similarly characterized by dynamism and interconnected subsystems. For example, in project management, the relations between tasks (in the form of preconditions) is used to identify critical paths (Armstrong-Wright, 1969), which are defined as the sequences of activities for which, when one of their activities is delayed for some reason, the project's finish date will need to be pushed back. The critical path is a major factor to account for in any managerial decisions regarding planning. In teams, the critical path could likewise drive decisions on adaptation. However, methods that sequence work to optimize the critical path are normative in that they search for one optimal (typically minimum-time) sequence for performing the work.

Another domain that deals with dependencies between subsystems that each perform parts of the work (similar to multiple team members performing work) is supply chain management. A supply chain is defined as "a network of connected and interdependent organizations mutually and co-operatively working together to control, manage and improve the flow of materials and information from suppliers to end users." (Christopher, 1992) Thus, "organizations" can be seen as agents of a team. "Flow of materials and information" are the product of a supply chain, similar to the results or outcome of work (changes in the work environment) in a team. Although supply chains generally operate and adapt on slower time scales compared to teams in complex work domains, the core principle of supply chain management is— just as a team that adapts it work strategies— to create a match between supply and demand.

Lead time in supply chain management is the longest duration sequence of tasks (similar to the critical path in project management), and is, together with the uncertainty of demands, central to the selection of appropriate supply chain strategies (Christopher, 1992). Different strategies then vary in the degree to which they create buffers for future demands and whether they deliberately optimize far into the future or focus on quick responses to new demands.

The concept of logistics system dynamics has been of interest for synchronization of supply chains, where the effects of design parameters such as the organizational structure, policies and time delays (in decisions and returns) on a chain's performance has been analyzed using dynamic simulation. The results from this dynamic modeling showed that small disturbances can have ripple effects on the rest of the supply chain.

2.2.3 Summary

This dissertation focuses on conceptual design of teams, particularly the determination of team composition, work allocation and teamwork modes. Making design choices on these factors is challenging because it is impossible to foresee all possible situations the team can encounter, yet one still wants to support adaptation. A distinction is drawn between normative and formative design methodologies. Normative design methods prescribe an optimal way of doing the work, but in practice such an ideal strategy falls short in actual context. Formative methods inform the designer, but allow the user to finish the design as appropriate in context. Another distinction is drawn between static and dynamic analysis methods. Historically, most formative methods are static and qualitative. In related domains, more dynamic analyses have been explored.

2.3 Human-Robot Teams

This section elaborates more specifically on the characteristics and design of human-robot teams, as opposed to the more general theme of teams (both human-human and human-robot) discussed so far. Specifically, the question that is addressed here is: What are additional factors that one needs to account for in the context of human-robot teams?

2.3.1 Changing Role of Technology

The role of technology in complex work domains has changed significantly in the past decades. Research in human factors initially focused on human-machine interaction, with technology being used by human workers as tools to conduct their work. Later developments led to the perspective of technology as a prosthesis to human workers, with

the aim of complementing human capability. With the advance of computational power, from technology-driven research domains, there has been a push towards autonomy as the ultimately goal for technology. However, many have argued that in a system, no agent should or could operate truly autonomously (Bradshaw, Hoffman, Woods, and Johnson, 2013); instead, technology needs to act as a team mate, working alongside human workers and effective interacting with the rest of the team. With such a perspective, design of human-robot teams can learn from human-human teams, and there have been several studies that have taken human-human teams as examples for how to design human-robot teams (Demira, McNeeseb, and Cookea, 2018; Shah et al., 2007; Sierhuis et al., 2003). With technology as a team mate, embodied forms of technology such as robots also evolve from systems that are mainly used as tele-operation devices to team members that interact with human team members both cognitively and physically.

2.3.2 Challenges to Human-Robot Team Design

One of the biggest challenges to the design of human-robot teams as opposed to human-human teams is that, compared to human workers, technology is rigid. Rigidity in a team member's behavior does not jive with characteristics of effective teaming that emphasize adaptation and flexibility to achieve resilience. To allow agents to respond and adapt to changing work demands, an allocation of work should support humans in responding to context, and should foster predictability to allow the agents to create stable work environment and manage uncertainty (Feigh and Pritchett, 2014).

A further challenge is that technology cannot be held legally accountable for the outcome of a task. Thus, when allocating work to a robotic agent, any responsibility for the

work remains with a human worker. Woods described this authority-responsibility double-bind, and its consequences when the double-bind is broken (Woods and Hollnagel, 2006). Thus, when tasks are allocated to robots, only authority for a task can be allocated to a robotic agent. As responsibility remains with the human, the operator is required to continuously or intermittently verify the robot's correct operation through monitoring its status, and also needs to be ready to intervene when the robot approaches or violates its operational boundary conditions.

Because of the two factors described above (rigidity and inability to bear responsibility), and because cognitive capabilities of robots are, to date, very limited (particularly when it comes to adapting work strategies to respond to context), robots require frequent human input into judgment and decision-making needed to execute their tasks. In the design of a human-robot team architecture, questions associated with directability include how human team members will direct robotic aids and how a human-robot team can shift fluently from one strategy to another (Woods, Tittle, Feil, and Roesler, 2004).

The degree to which a human operator is required to contribute to robotic operations is dependent on the capabilities of the robot. A robot with many cognitive capabilities (in that it can independently and reliably judge information and make decisions) may require input only for the highest-level goal specification (e.g. the human providing input in the form “perform this scientific experiment”), whereas a robot with few cognitive capabilities may require more frequent input to specify/correct its next few actions (e.g. the human providing input in the form “drive 10 ft forward, then turn 40 degrees left”). Similarly,

limited physical capabilities of a robot might require the human to intervene frequently (e.g., a rover that gets stuck in sand).

Woods et al. (2004) highlight three forms of human direction: plan-based direction in which contingency plans are made beforehand, constraint-based direction in which the robotic aid acts autonomously but operators direct the robot through setting bounds and constraints, and intent-based direction in which the intent behind a plan is communicated and any adaptation takes into consideration this intent. Likewise, (Fong, Zumbado, Currie, Mishkin, and Akin, 2013) describe how the control mode determines the input-relationship between the operator and robot. In space operations, for example, the two most frequently used control modes are direct teleoperation and command sequencing. In direct teleoperation, the operator provides continuous input and essentially performs work through the robot. With command sequencing, the operator provides commands beforehand, which the robot then executes semi-autonomously.

Finally, current-day robots generally take longer than human agents to complete their tasks (Fong et al., 2013), with typical time multiplier factors in the order of 1.25 to 6 (Akin, Hedgecock, and Sorenson, 1989; Singer and Akin, 2009). Thus, delegating dependent actions to robots might be undesirable when it will delay subsequent actions for other agents.

The main point is that off-loading tasks from humans to robots is not simply a matter of re-allocation; instead, it changes the role of the human from an executer to a supervisor of a task. With it, additional teamwork is created for the human to verify the robot's correct operation and provide necessary input, all while constrained to the robot’s potentially rigid

operations. The overhead of required monitoring and direction are only aggravated when, or more accurately, because robots are not as predictable as human team members, making human-robot coordination (through interweaving of human and robotic activity) difficult.

Indeed, there are many examples of ineffective human-robot teaming in which the introduction of automated or robotic technologies inadvertently limits the human worker in adapting to changing work demands. This results in "work-arounds" by the human. In such work-arounds, the delegation to robotic agents will deviate from what was envisioned by the designers, or it might not even occur at all, where the latter case is illustrated by problems with the disuse of automation (Kirlik, 1993).

2.3.3 Models for Human-Technology Integration

There are many models and methods available for the conceptual design or integration of humans and technology in sociotechnical systems, each with a different focus. Here, I mention the most influential methods, focusing on the design questions of team composition, work allocation, and interaction modes.

CWA is perhaps one of the most influential and also most comprehensive frameworks for the design of sociotechnical systems, as it provides a holistic analysis of factors to consider in human-technology integration. Many of its models such as the abstraction hierarchy, the decision ladder and the SRK taxonomy are used as standalone analysis methods. The original CWA publications have since been followed by many derivative methods and concepts, which apply the formative approach of analyzing constraints and affordances to different domains (e.g., Ashoori and Burns, 2013; Kilgore, St-Cyr, and Jamieson, 2008; Naikar, 2006).

Other methods, specifically for allocation of work between humans and automation, have focused on agent capabilities as a determining factor of effective work allocation. The most classic example is the original Fitts' list (Fitts et al., 1951) that describes what machine and humans are each better at. Later approaches include the levels of automation paradigm for determining what the machine will do and what the human will do based on a descriptive taxonomy of various forms of joint decision-making, ranging from the "human does it all" to the "machine does it all." (Parasuraman, Sheridan, and Wickens, 2000; Sheridan and Verplank, 1978). This taxonomy has, however, been criticized for being too coarse, and overly simplifying the problem. Others have criticized the taxonomy for being "summative," whereas formative methods are needed to guide design (Johnson, Bradshaw, and Feltovich, 2019).

A more recent design approach in the human-robot teaming community is co-active design (Johnson, 2014). This approach stresses human-robot "interdependence" as a central theme to the design of human-robot teams, and explicitly identifies the observability, predictability and directability requirements necessary to coordinate between interdependent agents. Coactive design considers all work in a team to be joint, where multiple agents possess different sets of capacities that are required to complete a task. Depending on how multiple agents work together to complete all capacities of a task, OPD requirements can be identified. By identifying these requirements as inherent characteristics of work strategies, the required coordination and impact of strategy change on team members can be classified and accounted for.

Pritchett, So Young Kim, Kannan, and Feigh (2011) introduced the idea of computational work models, which are based on the principles of CWA, but allow

designers to simulate or compute any emergent effects of their design decisions. Through simulation, one is able to explicitly assess the dynamics of any constraints. Computational work models have been applied in the aviation domain, evaluating human-automation function allocation on the flight deck (Pritchett, Feigh, Kim, and Kannan, 2014), and air-ground function allocation in air traffic control (Pritchett, Bhattacharyya, and IJtsma, 2016). In these applications, they focused specifically on the effects of aircraft dynamics as a driving factor of a team's work.

Thus, there is a variety of methods available for the design of teams, each with its own focus. Most of the existing methods are static, in that they do not account for dynamics of the work. Likewise, there are several formative methods available, but there is still a tendency of designers to employ simpler (but coarser) normative methods such as levels of automation.

2.3.4 Summary

The role of technology in complex work domains is changing from being used as a tool into being viewed as team members that work alongside humans as "agents" of their own. This comes with new design challenges associated with cognitive and physical interaction in teams consisting of humans and technological agents. Challenges include the rigidity of technology compared to humans, leading to potential issues with team adaptation; technology being unable to bear responsibility for the outcome of work, creating requirements for verification and cross-checking; and technology needing direction from human operators. Thus, off-loading tasks to technological agents is not simply an act of

delegation; instead, there is often significant overhead associated to interact and coordinate with a technological agent.

Traditional methods for the integration of humans with technology include CWA and the levels of automation taxonomy. CWA has made an impact by arguing for formative analysis of dependencies and constraints. Levels of automation is popular for its more systematic approach but is criticized for simplifying the problem and for its summative (as opposed to formative) approach. Co-active design is a more recent approach which has had success in the design of flexible human-robot teams. Finally, computational work models have been proposed to simulate emergent effects of design decisions. Further developments, however, can combine attributes of these earlier frameworks, striving for formative, systematic, and dynamic methods that can help inform designers in conceptual design decisions.

2.4 Example Work Domain: Manned Spaceflight Operations

Throughout this dissertation, case studies are used to illustrate the concepts. The domain of interest for these case studies is manned spaceflight operations. This section briefly discusses the main challenges to manned spaceflight operations from an operational design perspective.

What makes the design of operations for future manned space missions so challenging? This section describes five challenges. Three of them apply to past and present operations: (1) High uncertainty in the work domain, (2) sparsity of manpower and resources, and (3) lack of accurate testing environments. For future missions, there are two additional challenges: (4) communication delays and (5) astronauts working effectively with robots.

2.4.1 Operational Challenges to Past and Present Missions

The space environment is harsh and unpredictable and there is considerable risk. Thus, to maintain safety of the astronauts and the mission, an effective concept of operation requires managing the inherent uncertainty and associated risk by being robust to disturbances yet flexible enough to adapt to unforeseen circumstances. Additionally, in doing so, the concept of operation must account for sparsity in resources, as payload restrictions limit the number of astronauts and tools, and the amount of consumables, which can be transported to space. Consumables such as oxygen and power limit the total number of tasks an astronaut can execute consecutively. Artifacts or objects can likewise only be used by one astronaut at a time. Resources may need to be transported from one work location to the other, depending on when and where they are needed. Altogether, managing these constraints can create considerable overhead that needs to be accounted for in the design of the concept of operation (Looper and Ney, 2005).

Unlike in other safety-critical domains where there might be similar challenges, such as aviation and the nuclear industry, space exploration inherently is a first-of-a-kind operation and there is therefore little past experience to rely upon in the design of concepts of operation. Moreover, it is difficult, if not impossible, to accurately emulate the space environment on Earth, making the testing of potential concepts of operation a challenging and costly undertaking.

The near-Earth missions of the past decades have demonstrated that these challenges can be managed. The concepts of operation for these missions are, however, heavily influenced by them. They are modeled after a military command structure in which the

operations are centered around the Mission Control Centers (MCC) on Earth. Compared to the spacecraft, the MCC has effectively unlimited resources, manpower and expertise of the many systems and aspects of the mission. In such "ground-centric" operations mission control bears responsibility and has decision-making authority (Diggelen, Bradshaw, Grant, Johnson, and Neerincx, 2009). Consequently, the astronauts have little independence and rely on the support from ground.

In a discussion of voice loops as communication aids for mission control, (Watts et al., 1996) provide a breakdown of ground-centric operations of ISS missions. In the control center, teams of engineers monitor the spacecraft, robot systems and astronauts continuously, 24 hours a day. A hierarchical set-up of these teams allows for efficient coordination and communication between teams. The front-room team is the coordinating team in this organization. The highest in the hierarchy in this team is the flight director, bearing responsibility for all mission-related decisions. About twenty flight controllers, who are responsible for the various systems involved in the mission, support the flight director. The director and controllers are situated in the Flight Control Room – or front room – allowing them to communicate directly with each other and get a full picture of the various systems and their relations.

The controllers are assisted by teams of engineers in the backrooms, providing detailed support for each of the subsystems. These backroom teams have access to more detailed telemetry data and can assist in monitoring, diagnosis and re-planning in case of anomalies. Each of the teams has authority and responsibility for a particular subsystem allocated to it. Built-in cross-checking and communication protocols between the backroom, front-room and the astronauts further foster robustness of the operations.

2.4.2 Operational Challenges to Future Missions

In addition to the challenges for near-Earth spaceflight discussed in the previous subsection, future missions will likely face additional challenges associated with outer space. The two main challenges for outer space missions are the communication delays that demand a shift in the locus of control and the anticipated need for automation and robots working together with astronauts.

To illustrate the challenges associated with communication delays, the operational system of space missions can be described in terms of the locus of control and the locus of information (P. J. Smith and Mccoy, 1999). The locus of control indicates where the main decision-making authority is located within a system, whereas the locus of information indicates where the relevant information for decision-making is located.

In current manned space flight operations, the locus of control lies with the MCC. The locus of information, however, is situated mainly with the spacecraft as where the relevant data is known or measured. In near-Earth operations there is significant bandwidth available to transmit real-time data from the spacecraft to the MCC. However, in future missions farther from Earth communication delays between Earth and the spacecraft can be on the order of minutes. This makes the locus of information a critical factor in allocating the locus of control because, to reliably perform the decision-making tasks associated with the locus of control, one needs access to real-time information.

Additionally, collaboration between team members involving significant latencies can affect the natural communication between the agents (Fischer and Mosier, 2014). Even when a task does not require real-time information, there are two challenges for time-

delayed communication. First, the coordination of the communication process (e.g., the timing of speaking turns) becomes more difficult as part of the communication may become asynchronous. Second, the maintenance of a common understanding and situation awareness is challenging. Moreover, if there is a misunderstanding there is no immediate feedback by which to resolve it.

Thus, in concepts of operation for future missions, the locus of control needs to be shifted to the spacecraft to coincide with the locus of information. The MCC may stay involved in some tasks that do not require real-time communication, but protocols need to be in place to manage the time-delayed communication. Shifting the locus of control requires a significant overhaul of existing operations, with new solutions for the challenges associated with robustness, flexibility, and resource sparsity discussed in the previous subsection. It is also the main impetus for designing new automation and robotic support to assist the astronauts.

With the shift of locus of control, the number of tasks to be performed by the crew in space will increase considerably. In early planning reports for a manned mission to Mars, NASA and ESA have argued for a crew size of six (Drake, 2009; Horneck et al., 2003), but there are also studies that propose a crew size of four, where two teams of two astronauts meet up at Mars (Salotti, 2011). To off-load some of the astronauts' workload, NASA and other space agencies envision the increased use of automation, as well as robots. However, as robots will be working alongside astronauts, it raises challenges with how astronauts and robots can fluently work together in the context of the inherent abilities and inabilities of both astronauts and robots.

Typical capabilities of robots include the repetitive tasks that are considered "dull, dirty, or dangerous" and do not require significant cognitive capabilities (Elfes et al., 2008; Fong et al., 2013). The foreseen benefits of the use of robots in space are, first, that delegating tasks to robots will free up capacity for the astronauts to work on more cognitively demanding tasks. Second, as robots are better equipped to operate in harsh space environments than humans are, robots can be used to perform extra-vehicular tasks that would otherwise pose significant risk if the human were to execute these tasks. As an example, robots could do a significant portion of the routine maintenance on the outside of the spacecraft or do early reconnaissance of a planet surface to find suitable landing spots.

NASA and other space agencies have been investigating a wide variety of robotic agents to take over work from human agents. Example projects of space robots that are currently or foreseen to be employed on manned space missions are the Robonaut 2, a humanoid robot that is being tested on board the International Space Station (ISS) (Diftler et al., 2011), the Astrobee, a free-flying robot that can monitor science experiments and do simple manipulation tasks (Bualat, Barlow, Fong, Provencher, and Smith, 2015), and the Space Station Remote Manipulator System (SSRMS) or Candarm 2, a robotic arm used for maintenance and assembly tasks on the ISS (Hiltz, Rice, Boyle, and Allison, 2001).

2.4.3 Summary

Challenges for the design of operations for the manned spaceflight domain include significant communication latencies, the limited availability of agents (astronauts and robots) and resources, the unpredictability of the space environment (inherently, space exploration is a first-of-a-kind operation) and differences in human and robotic capabilities.

To design effective teams including both humans and robots, there is a need for objective methods for the evaluation and synthesis of work allocation, taking into account the aforementioned challenges and factors.

2.5 Summary of Identified Gaps

Based on current state of the art discussed in this chapter, effective teams in complex work domains maintain safety and efficiency as continuous processes of adaptation, making them resilient to changing work conditions and demands. Much of the ability to adapt stems from the expert worker's contextual control, changing behavior to maintain performance under a wide variety of conditions. An additional challenge when working in teams is that work, including adaptation, needs to be coordinated with other team members, human or technological.

Good design supports adaptation, as well as effective coordination between team members to assure efficient use of time and resources. Existing design frameworks have proved the value of formative over normative design, wherein design does not overly prescribe how the work ought to be done, but instead aims to support expert human workers in "finishing the design" as appropriate in context.

Much progress has been made in developing such formative methods, as demonstrated by the methods discussed in this chapter. Two remaining gaps can be identified, however. First, in the conceptual design of teams there is a need for more systematic methods that can help the designer assess the effect of design decisions associated with team composition, work allocation and interaction modes. CWA has laid the groundworks of identifying and reasoning about constraints and affordances that might drive the behavior

of a team, but its methods are least systematic and, as acknowledged by the Vicente (Vicente, 1999), are not generalizable beyond the case studies presented in his publications. Later models have expanded on these ideas, but further systematicity is desired for providing formative and comprehensive insight into the effects of team composition, work allocation and interaction modes on team adaptation. Such formative analysis of work strategies in teams would be useful for better understanding how adaptation in a team is affected by these design decisions.

Second, existing methods for conceptual design phases do not explicitly account for temporal aspects of work in a team. However, effective adaptation and coordination are inherently temporal concepts. Therefore, having objective insight in how a system would behave dynamically can benefit designers in better understanding the effects of their design choices. Evaluation of the dynamic effects of constraints and dependencies of work within a team cannot be done with most existing methods for team design, but other fields have used simulation to evaluate dynamic effects in their systems. Thus, inspiration can be drawn from related domains in which dynamics are explicitly accounted for.

CHAPTER 3. COMPUTATIONAL MODELS OF DYNAMICS OF TASKWORK

In teams that are adapting to changing work demands, the interweaving of activities is a major aspect of coordination. This adaptation and coordination of work in a team have significant temporal aspects, yet existing methods of work analysis have applied static work models that cannot account for the evolving dynamics of the work itself, and the coordination and synchronization it requires within a team. Thus, there is a need for methods that can provide insight into the dynamic and temporal aspects of a team's work.

In this chapter, a new approach is laid out that focuses on the work to be conducted in a team, its constraints and dependencies, and how these shape the team's temporal behaviors. Analysis of these work dynamics, through the use of computational work models, can provide valuable insight into how constraint and dependencies in the work and work domain drive behavior in socio-technical systems. The aim of this approach to provide designers of team with insight into how design decisions associated with work allocation and interaction modes impact the interweaving of activities in teams.

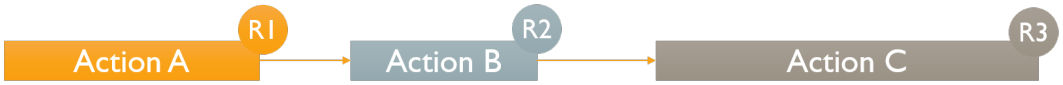
Section 3.1 discusses the concept of work dynamics. This is followed by a discussion in section 3.2 of how one can formally model work in a team and identify dependencies that affect work dynamics. Section 3.3, then, discusses how one can evaluate work dynamics through the simulation of computational work models. To illustrate these concepts, section 3.4 contains a case study, followed by a concluding discussion of the implications of this type of dynamic analysis of work in a team.

3.1 Work Dynamics

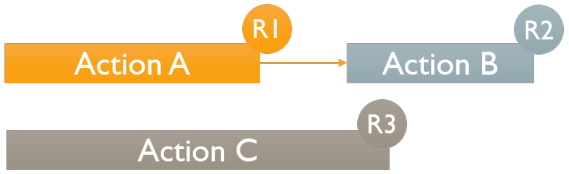
Situated work is intricately connected to the work environment (Pritchett et al., 2011). The work manipulates the work environment, and the work environment in turns drives and constrains the work. Work can be characterized by these dependencies (and interactions) between the work and the work environment through flows of information and through manipulation of physical objects or artifacts. As the work is performed, this interaction determines what can be executed and when. As the work progresses, patterns emerge in the way information and artifacts are shared between different parts of the work. How these patterns evolve as a consequence of interplay of dependencies in the work is defined here as work dynamics.

To illustrate, consider three actions, actions A through C, which can be dependent on each other in a number of ways. For example, Figure 3.1a shows that, when action B requires the output of action A, and action C requires the output of action B, there is an implicit sequence of actions that the team must abide by. Figure 3.1b illustrates how, when action C is independent of actions A and B, action C can occur in parallel with the other two actions. Further, Figure 3.1c represents how, when action C uses the same artifacts as actions A and B, there is another implicit requirement where none of the actions can occur in parallel. Thus, Figure 3.1 illustrates how these dependencies result in different feasible action sequences.

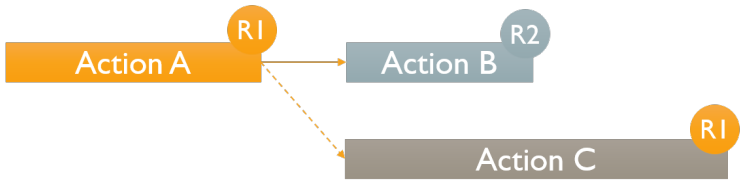
Even in this simple example it is fairly obvious how the action sequences follow from the constraints and the work; in more complicated work models, with actions being executed by multiple agents, the resulting work patterns are difficult to predict from just a



(a) Actions B dependent on A, and action C dependent on B



(b) Action C independent from actions A and B



(a) Action A and C implicitly dependent through artifact R1

Figure 3.1: Illustration of dependencies resulting in work dynamics.

static analysis alone. For example, when work is distributed over multiple locations, and the locations of agents and resources need to be managed, the dependencies in work drive the required traversals of agents. Thus, when actions A, B, and C all use the same physical resource, but take place in different locations, these dependencies require additional actions to bring the physical resource from one action location to the other.

Moreover, these dependencies affect the coordination requirements in a team. For example, in the previous example with three actions, when action B is allocated to a different agent then actions A and C, these two agents need to wait for each other and therefore coordinate their work. Thus, these constraints determine to a large degree how

long each agent is busy versus waiting (idle) for other agents to complete their work. Such coordination requirements then drive the temporal behavior of the team, including the temporal affordances that are available to the team.

Other factors that determine the temporal patterns of work include dynamic processes taking place in the work environment. For example, in air traffic control the dynamics of aircraft are a constraining factor in the timing of the air traffic controller's actions. Similarly, in the nuclear energy domain, the dynamics of the nuclear reaction determine when certain action sequences are appropriate. In teams, the spatial management of agents and resources require locomotion of agents that, as dynamical processes, can affect timing of activities (e.g., required traversals to fetch resources).

As discussed in the literature survey in the previous chapter, related domains such as project management and supply chain management have investigated how dependencies between components result in temporal behavior. In the cognitive systems engineering, these dependencies are certainly of interest, but no systematic and dynamic analysis has considered the temporal effects of these dependencies within a team. However, insight into how work dynamics play out as a result of decisions associated with team design can be valuable in early design phases. This chapter discusses an approach for performing dynamic analysis to explicitly evaluate these effects.

3.2 Model of Work and Dependencies

The first phase of CWA is to conduct a Work Domain Analysis (WDA), which identifies the constraints and affordances that a system (or a team of workers) must abide by. To conduct a WDA, the Abstraction-Decomposition Space (ADS) provides a basic framework

(Rasmussen, Pejtersen, and Goodstein, 1994). An ADS of a work domain describes the system at various levels of abstraction along the vertical axis, and various levels of decomposition along the horizontal axis. Each elements along both dimensions comprises a full description of the system. When it was introduced, the decomposition dimension was common in models such as the Hierarchical Task Analysis, but the abstraction hierarchy was added to also model the means-end relationships that drive system behavior, ranging from conceptual definitions of the purpose of a system to its physical representation.

Thus, at the highest level of abstraction, a description of the system consists of its purpose. At the lowest level of abstraction, the description entails the physical and tangible elements that comprise the system. Levels in the hierarchy are connected through means-end relationships, where a lower level is a means for achieving any of the elements in the upper level. Commonly, as one goes down the hierarchy one can ask the question "how?", whereas if one goes up one can ask "why?" With an ADS, designers can identify any constraints and affordances that a socio-technical system needs to abide by, irrespective of how the work is performed and which agents are involved.

To model work in a team, a similar approach can be used: the work can be characterized at different levels of abstraction, each providing a complete description of the team's work, see Table 3.1. At the highest level of abstraction, a description would include the purpose(s) that the team is being designed for. A purpose can be abstracted down into abstract functions, describing the values and priorities for the team. At the middle level, one can describe the team in terms of its work processes, as means through which the team can achieve its values and priorities at the level above. At the physical functions level, the team's work can be described in terms of activities that a team can perform. Finally, at the

Table 3.1: Five levels of abstraction to describe a team's work domain (Rasmussen, Pejtersen, and Goodstein, 1994; Vicente, 1999).

Functional purposes	Purpose for which the team is designed
Abstract functions	Values and priorities for the team
Generalized functions	Work processes of the team
Physical functions	Activities that the team can perform
Physical forms	Tangible aspects of the team's work and their characteristics

lowest level of abstraction, the team's work can be described in terms of the physical artifacts and information that is manipulated as the team performs its activities. Thus, this latter level describes the most tangible aspects of the work.

Work models capture the two lowest levels of an ADS with their focus on actions, which are descriptions of work's interaction with and on the environment, and resources, which represent the information and physical entities in the work environment that actions interact with. Thus, contrary to work models that account only for the decomposition dimension, this work modeling approach also explicitly models the physical shapes and characteristics of the work environment. Including these representations allows for deliberate analysis of how these elements drive the dynamics of the work, accounting for the interactions between the work and the work environment.

Thus, the physical functions in an ADS represent the work functions or actions that a team can perform. Depending on the desired level of detail in the analysis, action modeling can be fairly high-level or detailed descriptors of the work or the environment. For the conceptual design of teams, in which the interweaving of the work of different team

members is of interest, it is useful to consider descriptors of work that are elemental enough to be executed by individual agents. Thus, high-level functions of the entire team can be decomposed along the horizontal decomposition-axis. Such descriptors of work, once formally assigned to an agent, then represent the interaction of an agent with the work environment.

For modeling work in a team, it is furthermore useful to split the physical form level into two concepts: physical and information resources. Physical resources (PR) represent artifacts in the environment that can be manipulated by and/or used for performing work. Physical resources need to be shared between team members, depending on how the work is allocated to the different agents. Examples of physical resources in manned spaceflight operations include tools and personal life support systems, but also galley ways of a spacecraft.

Information resources (IR) describe states of the work environment. For example, the speed of a vehicle or the location of an agent can be modeled as an information resource. These information resources can be standalone pieces of information that are "in the head" of individual team members, or "in the world," corresponding to views stemming from ecological psychology about "cognition in the world" of teams working in complex work domains (Hutchins, 1995). Here information resources might even be forms in between, such as artifacts that represent pieces of information. Additionally, a physical resource can have one or more information resources associated with it, specifying the artifact's characteristics. For example, a drill can be represented as a physical resource, with lower-level information resources that specify its location, its battery level and whether it is switched on or off.

Multiple heterarchic linkages can then be identified between the levels. To identify dependencies within the team's work arising from the work environment, three types of linkages need to be identified between actions and resources:

1. Actions can require physical resources (such as tools), described through use relationships. By definition, in this dissertation, physical resources can only be used by a single team member for a single action at a time. Thus, physical resources create constraints in what actions can be executed in parallel.
2. Actions can interact with information resources in two ways:
    - Actions require as input specific information resources, formalized as get relationships.
    - Actions serve to change the work environment, described as changing aspects of the environment state through set relationships where actions manipulate pieces of information as output.

These linkages are represented in Figure 3.2. A fourth type of linkage that is shown in Figure 3.2 is between physical resources and information resources. Physical resources are described by information resources that represent their states. Thus, an action can be linked to use the drill, but when it can also change the location of the drill, the action is also linked to set the information resource "drill location".

The dependencies between actions and resources can be used to identify several logical patterns or sequences in the work. When Rasmussen introduced the ADS, he noted that, as workers interact with the environment through their activities, they move through the

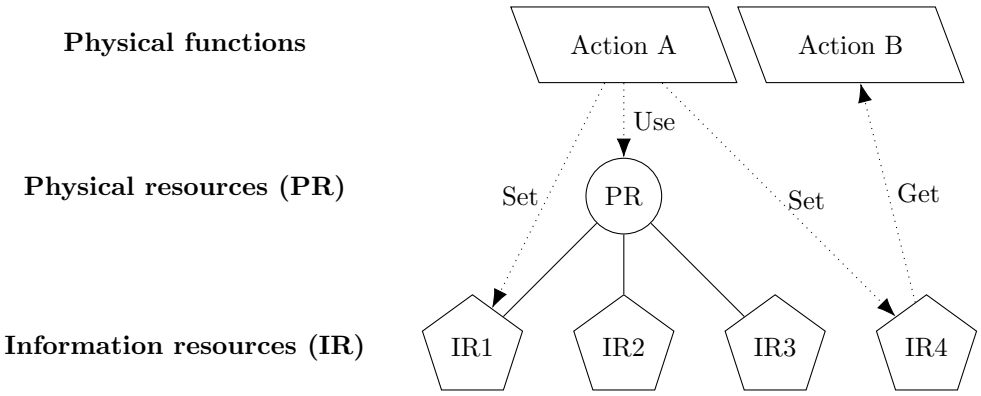


Figure 3.2: Dependencies between levels of abstraction in a team's work.

different levels of abstraction of the ADS (Rasmussen et al., 1994). Thus, as a team conducts its work, it moves from manipulation of the physical form of the work environment, through actions, which results in patterns of work that achieve goals.

This is similar to how Hollnagel decoupled the model of competence from the model of control (Hollnagel, 1993). Here, the description of the work at the various levels of abstraction in the ADS can be thought of as the model of competence for the team, i.e., what the team can do. The dependencies between actions and resources, then, provide insight in how agents could manage the sequencing of the actions in their model of competence while accounting for their dependencies. For example, in Figure 3.2, IR4 is input to Action B, where IR4 is set by action A. Thus, there is an implicit dependency between action A and B, where a logical sequence would be to first execute action A, followed by action B.

The modeling of dependencies between physical forms and physical functions forms the basis for the evaluation of work dynamics. These dependencies determine what actions can be executed, and when. As a team performs its work, these dependencies must therefore

be managed by a team. Dynamic analysis, then, can provide insight in how design decisions would impact the feasible action sequences for the team.

3.3 Simulation of Work Dynamics

For dynamic analysis of the effects of dependencies in the work, the actions identified in the previous phase, and their relationship to information and physical resources in the work environment, are modeled in a computational form. The simulation framework Work Models that Compute (WMC) provides the basis for the modeling and simulation to perform dynamic analysis of teams. The WMC computational simulation framework was developed to evaluate concepts of operation in dynamic environments (Pritchett et al., 2014). The framework relies on the same modeling constructs discussed in the previous section: actions, physical and information resources, and agents.

3.3.1 Computational Work Model

The computational form of the actions each specify which information resources an action gets (reading their value), some elemental calculation or conditional statements upon these representations of environment state, and which information resources the action sets (changing their value). Likewise, the work model defines which physical resources an action needs to perform the work. For example, the code for an inspection action gets the information resource "panel condition" to retrieve information on the panel's condition. The action model contains conditional statements to discriminate between two potential environmental conditions: (1) the panel is broken and needs repair, in which case this action sets the information resource "panel inspection results", or (2) the panel is in fine condition, and no further action is needed. Furthermore, this action uses the physical resource "panel

inspection tool," and thus during this action this physical resource is unavailable to other actions.

At their most basic, action models may employ very simple representations of how they use the resources they get to perform their work and how this work is then reflected in the resource values they set. Where the actions operate on dynamic environments, their models can also involve more involved internal representations of dynamics. For example, a simple action model moving a robotic arm might just set the associated information resource to the final position; however, where the dynamics of the arm might be continuously controlled or monitored by a human, or might obstruct the work of others as the arm passes through a location, the action model might instead include a dynamic model of how its position and velocity varies through time in response to command inputs. This was demonstrated in earlier work in the air traffic management domain, in which aircraft dynamics determined how and when pilots and air traffic controllers needed to interact (Pritchett et al., 2016).

When the simulation is started, the simulation's core action list contains the list of taskwork actions sorted by their respective predicted update times. The simulation proceeds by stepping through this list. Some dependencies within the collective work of the team arise out of the taskwork once it is executed within a specific scenario and work environment. Thus, as an action moves to the top of the action list, it is examined for whether it needs to engender other taskwork actions as required by dependencies. For example, during simulations the simulation framework recognizes when an agent's allocated actions require the agent to traverse from one location to another, or fetch a physical resource required for an upcoming action. In this case, the simulation will

automatically engender "traversal" and "fetch" actions. The simulation can also be scripted to insert faults or trigger failures by any agent to perform their actions correctly, where these events then require detection and resolution within the team.

Likewise, an action may need to be delayed due to unavailability of physical resources (e.g. a different action is currently using a required tool). All of these considerations can add new actions to the action list and change the sequence and timing of the actions' possible update time. Thus, the action list changes dynamically as the simulation is run, and the action at the top of the action list can only be executed once its dependencies and constraints are met. For this, the simulation framework passes the action model to the agent model allocated authority for its execution.

3.3.2 Agent Model

The framework can use any type of agent model that accepts calls from the simulation framework to execute actions during run-time. A perfect agent model will complete all actions, starting immediately upon their receipt and lasting for an expected duration. Easy extensions to the agents' models can further capture the dynamics they bring to the team collective behavior. For example, an agent may be flagged as only being able to perform one action (or a given number of actions) at the same time; in this case, an agent model can further delay when it executes its assigned actions according to which it prioritizes to do first.

Many additional performance considerations and variations are possible. For example, actions can be given a priority, which an agent model can then use to interrupt ongoing lower-priority actions to attend to incoming higher-priority actions. Another example of a

modeling consideration that could impact a team's work dynamics includes simple task-switching models, which can account for an additional time period for an agent to switch from one action to the other.

3.3.3 Metrics

These many potential dependencies and constraints inherent to a team's collective work are difficult to reliably predict without simulation. Existing methods for allocation of work, however, most often use a static analysis, risking that emergent effects will only become apparent in detailed testing once technologies, team compositions and procedures are sufficiently established that they are difficult to change. Thus, the benefit of using computational simulation in this methodology is that one can predict how these dependencies and constraints interact to create emergent effects in the various metrics of interest, providing objective, quantitative data on how these might affect the performance of a team.

To provide this insight, the simulation engine logs when each action is performed. Additionally, the simulation engine logs the transfer of information resources (as a measure for required communication) and exchange of physical resources (as a measure for required physical interaction), as actions executed by different agents share information and physical resources.

From these basics, one can compute a wide variety of metrics to gain insight in the efficiency of work, and how long agents are occupied and thus unavailable for other work or missions. Such metrics include, amongst others, mission duration, idle time and total time on tasks. Taskload measures quantify how busy agents are, both in the aggregate as

well as at any specific time. When the agent models do not impose a taskload saturation limit, such measures can identify when the default taskload imposed on any agent might create excessive workload.

3.4 Case Study I: On-Orbit Maintenance

This section describes the first case study in this dissertation, with the purpose to illustrate the concept of work dynamics. The first case study focuses on on-orbit maintenance of a spacecraft, where an effective team must interweave their collective work in such a way that the resulting work patterns promote time efficiency and reduces resource consumption, while maintaining overall safety.

Work efficiency on maintenance EVAs is of major concern. Looper and Ney analyzed work efficiency for EVAs onboard the ISS, and noted how just 7% of time is used for the actual EVA goals (Looper and Ney, 2005, 2006). The rest of the time is spent on traversal between locations, equipment preparation and other miscellaneous activity. They attribute this low efficiency to the fact that, early in the design of the ISS program, designers focused on design requirements for fulfilling the individual maintenance task, but the larger operational efficiency was not considered explicitly.

Dynamic evaluation of work dynamics in early design phases can be an effective way to gain insight and elicit requirements to allow a team to interweave the work and achieve efficient maintenance operations. Such analysis can help to guide the design in the earliest stages to support the EVA missions as best as possible. Furthermore, in future missions, communication delays associated with deep space will require astronauts to perform maintenance tasks without real-time support from Earth, further increasing the need for

careful analysis of how a team of astronauts would conduct such maintenance work in an effective way.

In this case study, the focus is on a hypothetical EVA mission in which astronauts are to inspect three panels on the exterior of a spacecraft (for example, the ISS). These panels are suspected to require replacement. Should the inspection find that the condition of a panel is indeed below the required specification, the astronauts are to immediately replace the panel with a new one.

An ADS describes the work of the team, shown in Figure 3.3. The upper-level mission objective is to maintain the spacecraft's integrity. This objective can be described in terms of three values and priorities: safety of the team, time efficiency and minimum resource consumption. At the generalized function level, four functions were included: airlock interaction, locomotion, inspection, and panel replacement. One level below, these generalized functions can be described through thirteen physical functions. The physical functions here were defined as elemental actions, reflecting the activities each occurring at a distinct time by single agents. The work can be broken down further along the horizontal decomposition axis, but for the demonstration of work dynamics this degree of decomposition is considered appropriate.

The lowest levels detail the physical and information resources. The physical resources include the tools needed for inspection of the panels and toolkits for replacement of broken panels. Additionally, there are backup components when a complete replacement is required. Information resources include attributes of the physical resources. Each physical resource has a location, panels have a condition, and tools have an attribute specifying

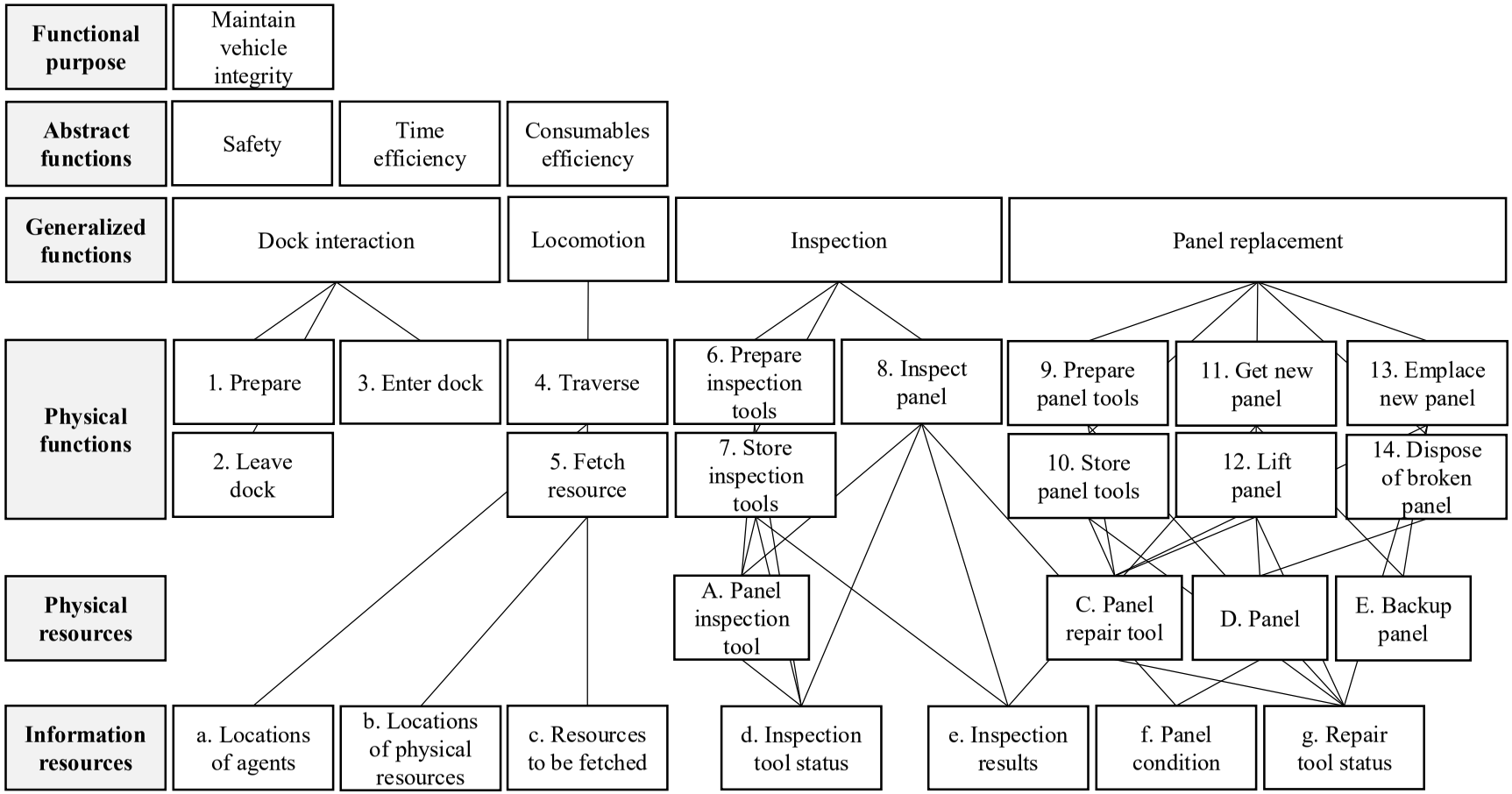


Figure 3.3: Abstraction hierarchy for the on-orbit maintenance scenario.

whether or not it is prepared and turned on. Other information resources (unrelated to physical resources) include the results of inspections (which can specify a list of the components that need to be replaced), and the location of agents. The ADS also shows the linkages between actions and physical and information resources.

Within this ADS, heterarchic dependencies were identified between the different levels. For example, between the generalized and physical function levels, dependencies denote the actions through which one achieves each generalized function. Then, the dependencies between actions and resources represent the use, get and set relationships discussed in the previous subsection. For example, "8. Inspect panel" uses the "A. Panel inspection tool", gets the "d. Inspection tool status" and the "f. Panel condition," then sets "e. Inspection results."

From the ADS, several patterns of actions can be identified. At a broad level, one first needs to inspect a panel before one can replace it, depending on the results of the inspection, as captured in the "Inspection results" resource. More specifically, when considering actions associated with the "Inspection" function, given that to inspect a panel one needs the panel inspection tool to be available and switched on, one first needs to prepare the inspection tools before starting the actual inspection. Likewise, after inspection, one needs to store the tools again.

Thus, inspection is a linear sequence of preparing the required tools, applying these tools and subsequently storing them away, as shown in Figure 3.4. Each action in this sequence requires the use of a toolset which will be unavailable to other actions at that time. It is assumed that each panel is in a different location, and thus traversal actions are

required to move the required physical resources and the inspecting agent from one panel to the other. This also means that inspection and repair actions for two different panels cannot be performed by the same agent simultaneously.

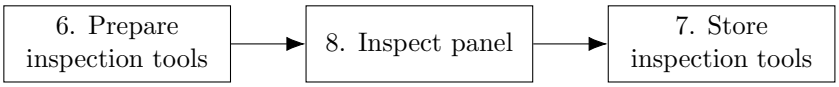


Figure 3.4: Sequence for inspection actions.

Likewise, replacement of a broken panel entails obtaining the required tools, removing and disposing of the broken panel, retrieving and emplacing a new panel from the inventory, and finally storing away the tools. The precedence relationships between these actions result in two sequences that are connected to each other, see Figure 3.5. The first sequence consists of getting the repair tools, removing the broken panel, and disposing of it (4.1 – 4.3 – 4.5). The second sequence consists of getting a new panel, emplacing it, and storing away the tools (4.2 – 4.4 – 4.6). The sequences are interconnected as emplacement of a new panel can only start once the broken panel has been removed. Furthermore, the actions 4.1, 4.3, 4.4 and 4.6 all require a toolset which cannot be used simultaneously by multiple actions.

The actions are modeled in WMC, together with physical resources representing the toolsets, the panels to be inspected and new panels for replacement. Each action is modeled as a small piece of code with some logical operators. For example, the action “Inspect panel” checks the information resources “Panel condition,” which is modeled as a Boolean. If it is false, it will set the resource “Inspection results” to the name of the panel that needs to be replaced. If it true, the “Inspection results” will be set to an empty string.

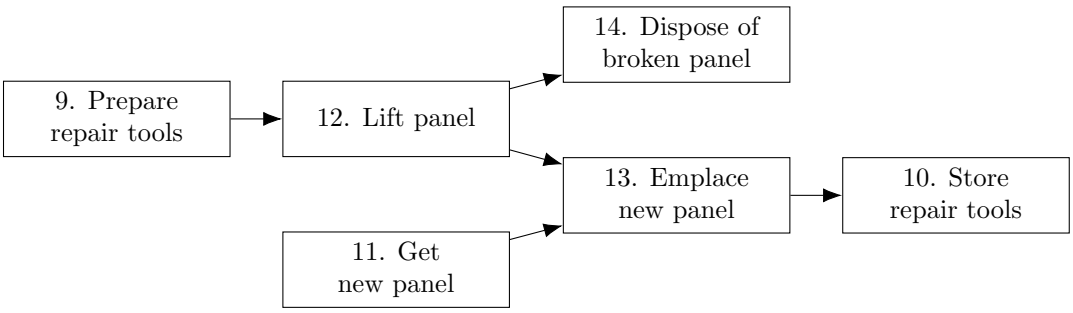
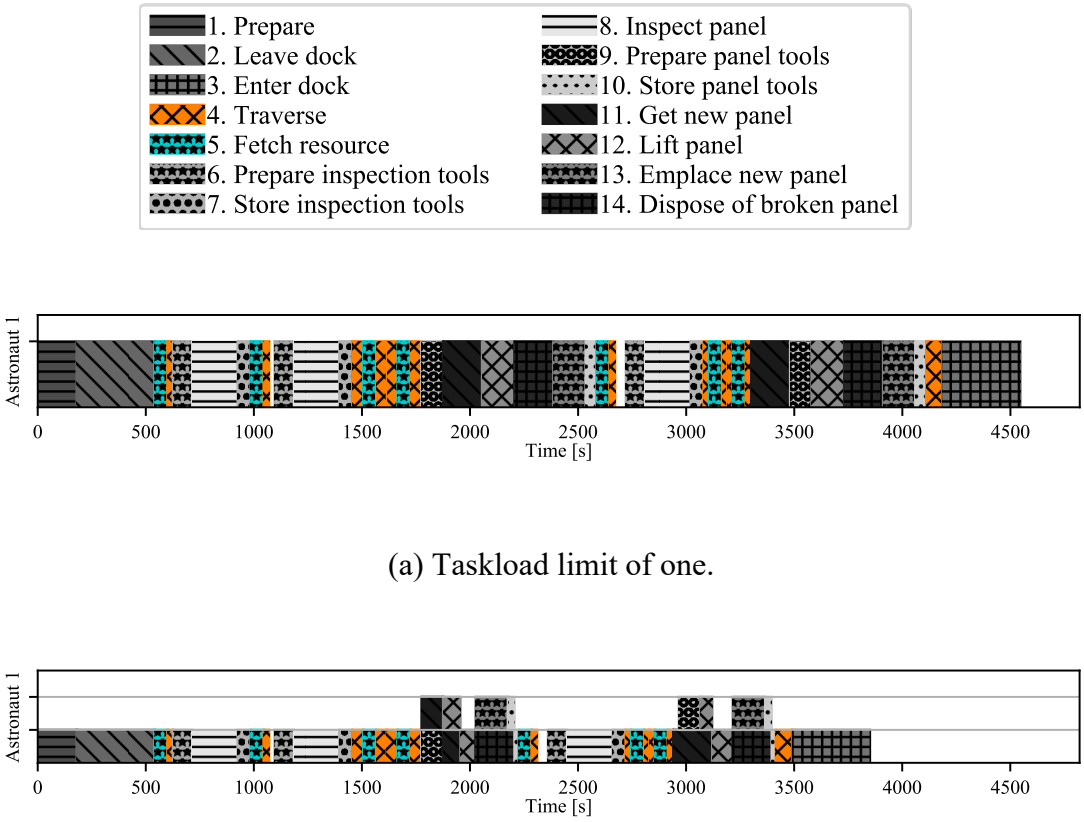


Figure 3.5: Sequences for repair actions.

The work model was simulated in WMC, which steps through the actions, taking into account the dependencies in the work, resources and agents when identifying when each action can be executed. The work dynamics can now be analyzed in two ways. First, we can set a taskload saturation limit for each agent and analyze the resulting work patterns. In such test cases, the work dynamics are governed by interactions between the limits of the agent model and the demands of the work model. Second, without a taskload limit in the agent models, the simulation results reflect the inherent taskload given to the agents, identifying possible workload spikes and drops as caused by taskload inherently imposed on each agent. Thus, the work dynamics are simulated with two different taskload saturation levels: the agents are capable of performing one (taskload limit = 1) or infinitely many (i.e., no taskload limit) actions at a time.

Figure 3.6 shows the resulting timelines of when each action can be executed, with a taskload limit of one (subfigure a) and no taskload limit (subfigure b). To emphasize how dependencies between actions and physical resources requires spatial management of resources and agents, the traversal and fetch actions have been highlighted in color.



(a) Taskload limit of one.

(b) No taskload limit.

Figure 3.6: Time traces obtained from the simulation of a single agent doing all the work.

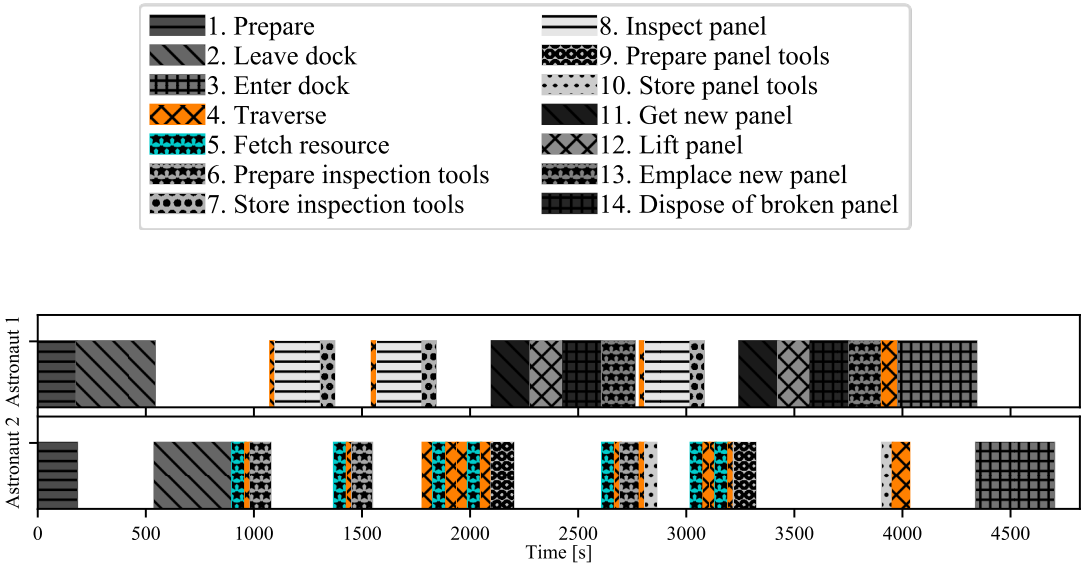
With a taskload limit of one, the resulting work pattern is one long, linear action sequence executed by a single agent and the ordering is determined by dependencies between actions alone (e.g., first inspection, then repair, etc.). At least four traversals are necessary to move the astronaut from the airlock to each of the inspection sites, and back to the airlock. Additional traversals are necessary when a panel requires replacement: now the astronaut must pick up a backup panel from the airlock area, as well as the tools required for replacement. In this simulation run it was assumed the astronaut can only carry one

physical resource at a time, but alternatively the astronaut could bring both the new panel and the repair tool to the replacement site in one traversal-and-fetch operation.

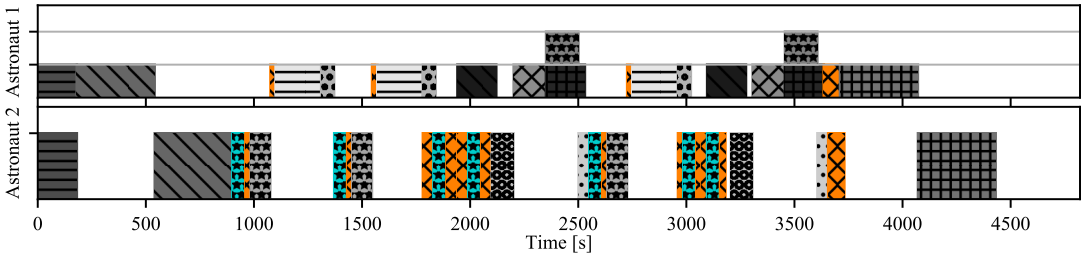
Without a taskload limit, the work pattern is solely the result of dependencies in the work and not influenced by any agent constraints. The pattern still reflects a more or less linear sequence, except that for panel replacement certain processes can be performed in parallel. For example, “Get new panel” can be executed in parallel with “Prepare panel tools” and “Lift panel,” as one would expect from the dependencies shown in Figure 3.5. Thus, such a run reflects how, if the agent is not a constraining factor, the work dynamics might impose higher taskload for certain parts of the mission.

When an additional astronaut is available to perform the work, the dependencies in the work will need to be managed by the astronauts through coordination, interweaving each agent’s work to abide by these constraints. To provide a simple demonstration of work dynamics in a team setting, a two-person astronaut team was modeled in WMC. The actions now need to be allocated to one of the two agents. For this demonstration, the second astronaut is assigned the actions associated with the fetching and preparation of tools, which includes the fetch action, and the “Preparation inspection tools” and “Prepare panel tools” actions. The resulting time traces are shown in Figure 3.7.

The dependencies between the actions result in interdependencies when these actions are allocated to different agents. Astronaut 1 now needs to wait for Astronaut 2 before he/she can continue with inspection and replacement of panels. Vice versa, Astronaut 2 needs to wait for Astronaut 1 to finish work before he/she can fetch or prepare the required tools. Additionally, now that two agents are involved, there are also two agents that require



(a) Taskload limit of one.



(b) No taskload limit.

Figure 3.7: Time traces obtained from the simulation of two agents interweaving the work.

spatial management, with associated traversal actions. Thus, there is more work to be performed by the team compared to just a single individual.

Moreover, the astronauts now share physical resources, which prevent certain actions from being executed in parallel. The clearest example of such a constraint is with the “Leave airlock” and “Enter airlock” actions. Here it is assumed that only one airlock is

available, with a capacity of one astronaut at a time. Thus, the astronauts need to wait on each other as there are leaving and entering the spacecraft. Adding an additional airlock, or an airlock that allows the astronauts to (de-)pressurize simultaneously would notably reduce the scenario time.

Furthermore, the data here suggest that the idle time can be reduced by allocating the panel replacement actions in a different way. Because Astronaut 2 is allocated the action "Store repair tools", he/she has considerable idle time near the end of the mission. Had this action been performed by Astronaut 1, Astronaut 2 could have returned to the airlock while Astronaut 1 finished the panel replacement.

Figure 3.8 shows the total time that agents were involved in the scenario, from their first action to their last action. Without a taskload limit, the work is completed more quickly, because certain actions can be performed in parallel as shown in the time traces. When comparing the two astronaut team to a single astronaut, the team completes the scenario in about the same duration as the single astronaut, but adding up both astronauts' time, the total time in the scenario is much higher. With more man-hours spent in the scenario, resource consumption is likely to be higher too.

The case study described here is for demonstration purposes, but the general principles can be followed in larger-scale scenarios that will be tested in later case studies in this thesis. As a final note, this case study demonstrated mostly dynamics due to constraints in the work itself, but external dynamics (in the work environment) that affect the appropriate timing of actions can also be accounted for by modeling these dynamical processes in the

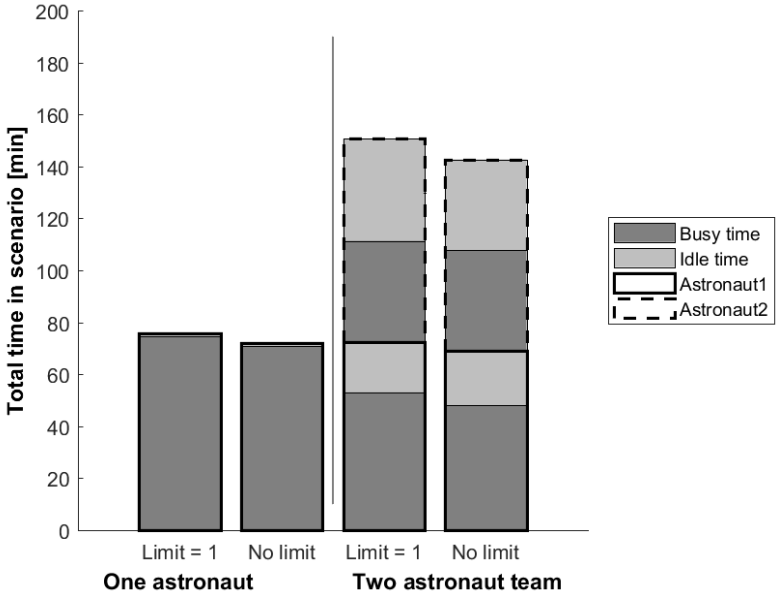


Figure 3.8: Time that agents are involved in the scenario, split into busy and idle time.

simulation framework. For example, a possible extension in case study II of this thesis is to consider rover dynamics as an additional driving factor of work dynamics.

3.5 Case Study II: Lunar Rover

The second case study in this chapter provides a more elaborate demonstration of work dynamics that involve continuous processes in the work environment. NASA and other space agencies are considering sending manned exploration missions to the lunar surface for geological research (International Space Exploration Coordination Group, 2018). For efficient exploration astronauts can use a rover to drive around, which has proven effective during the Apollo missions (Burkhalter and Sharpe, 1995). In addition, a rover could be used to drive semi-autonomously around the surface before or after human exploration (Deans et al., 2009), or in support of astronauts working on the surface (e.g., inspections,

surface exploration, fetching resources and/or returning samples to and from a lunar base) (Fong et al., 2008, 2010). Thus, a team of astronauts need to work with the rover and potential automated capabilities to fulfil mission EVA objectives.

Earlier work on designing human-rover teams has focused mostly on tele-operation. Stubbs, Hinds, and Wettergreen (2007) conducted observational studies of a team remotely operating a rover in the Chilean dessert and noted how, when shifting decision authority from the remote team to the rover systems, issues with maintaining "common ground" between the remote team and the rover shifted from a lack of necessary contextual information to a lack of transparency on what the rover was doing. Hooey, Toy, Carvalho, Fong, and Gore (2017) identified contextual factors that might drive operator workload for a conceptual lunar rover.

Following a brief literature survey on lunar EVA and Mars Rover operations (Hooey et al., 2017; Miller et al., 2017), as well as several informal discussions with space robotics and operations researchers, a work model was created in the form of an ADS, shown in Figure 3.9. Linkages between elements in this model identify constraints on the work. For example, "check temperature" is performed by getting the "subsystem temperatures" resource, and subsequently setting the "observed subsystem temperatures" resource. In turn, "observed subsystem temperatures" then would be input to "plan path for rover", creating precedence constraints that in turn identify feasible (or infeasible) sequences in the work.

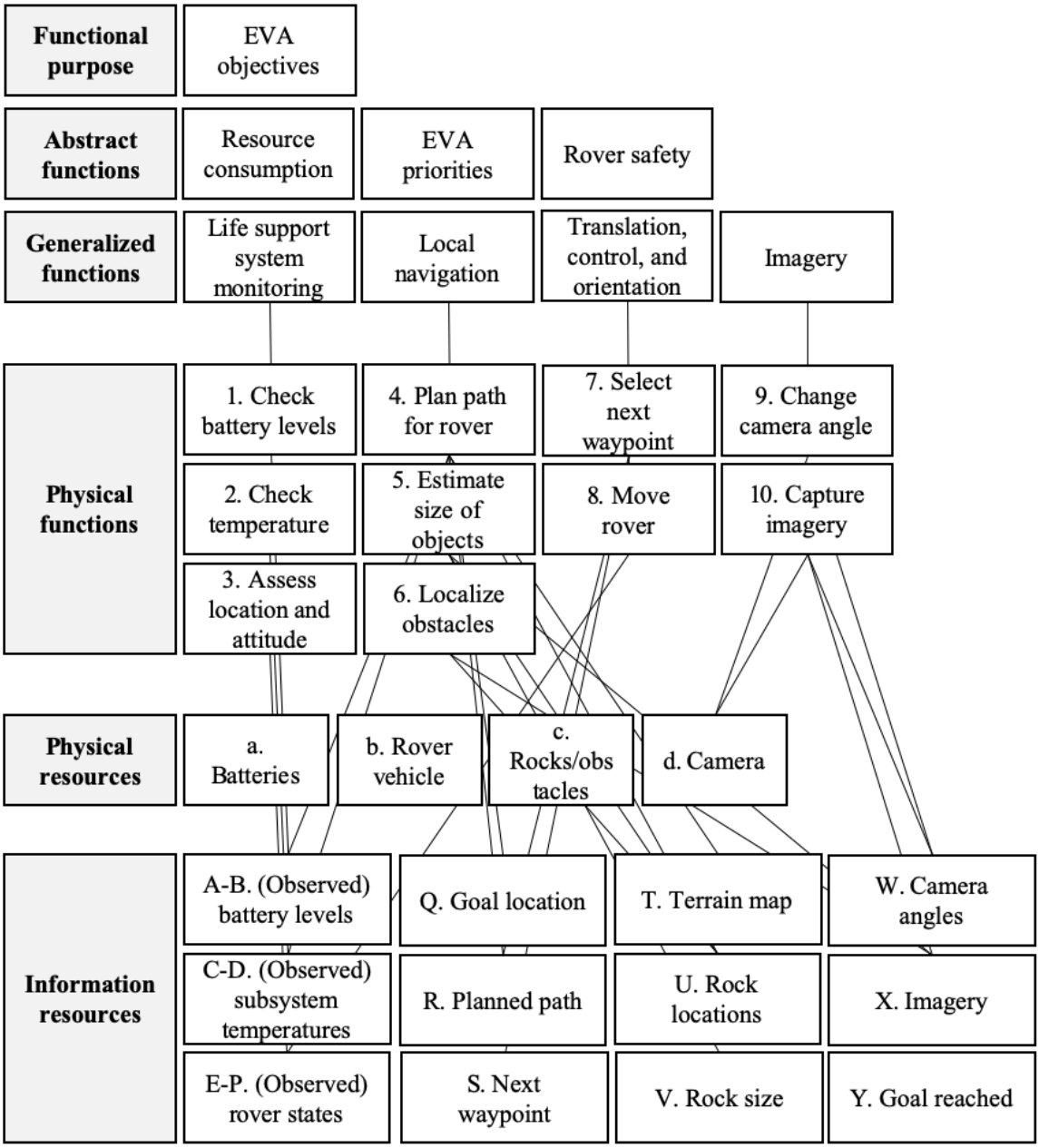


Figure 3.9: Abstraction-decomposition space of the work in a rover/astronaut team.

A computational version of the work model was created to evaluate the temporal dynamics of the work. From the dependencies in the work, it follows that the rover dynamics play a significant role in the work dynamics. By modeling these dynamics in detail, the analysis can capture the interaction of the continuously evolving rover states with the rest of the taskwork. Thus, the computational model of the action "MoveRover" contains a dynamic model of the rover, including simple controllers for speed and heading to have the rover drive to a waypoint, as specified in the information resource "NextWaypoint".

The longitudinal and lateral dynamics of the rover are approximated with a unicycle model:

$$\dot{v} = \frac{F}{m}$$

$$\ddot{\theta} = \frac{\tau}{I_{zz}}$$

Where v is the forward velocity of the rover, F is the forward force, and m the mass of the rover including any cargo or passengers. θ is the heading of the rover, τ the torque exerted by turning the wheels, and I_{zz} is the moment of intertia around the rover's vertical axis. The kinematic constraints are modeled as:

$$\dot{x} = v * sin\,(\theta)$$

$$\dot{y} = v * cos\,(\theta)$$

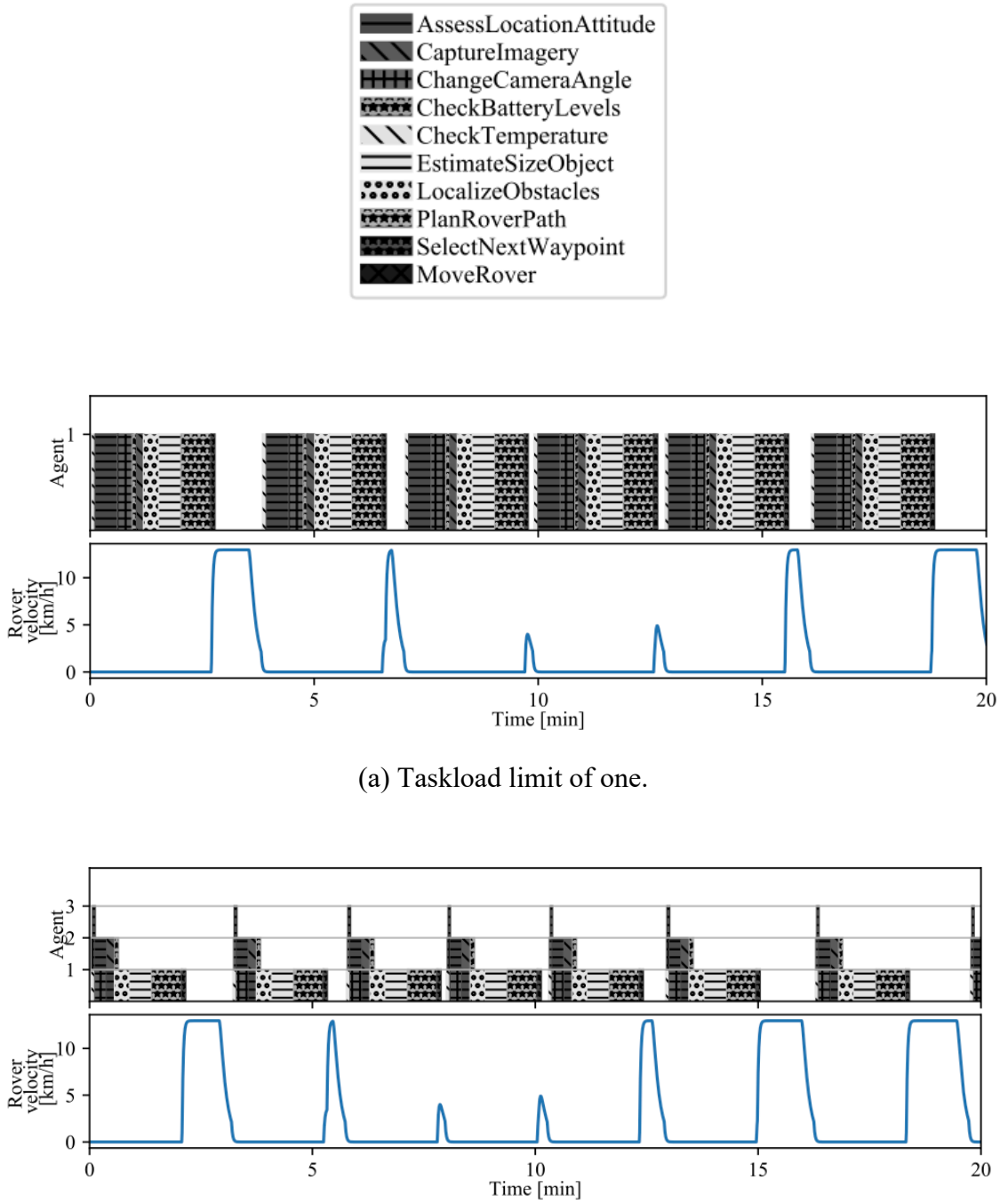
The parameters of this model are based on the Apollo Lunar Rovers (Costes, Farmer, and George, 1972). These rovers had a mass m of 708 kg (including two astronauts and their life support systems), and their maximum speed was set at 13 km/h. The simulation model is updated at a rate of 2 Hz. To control the position of the rover, proportional controllers were implemented, with inner-loop speed and heading controllers and an outer-loop position controller that directs the rover to a waypoint. It should be noted that more advanced dynamical models of the rover can be implemented. For example, as one progresses from conceptual design of the operations to more detailed design phases, the computational work model can be updated and refined to model the finer-grained dynamics.

A lunar terrain model (modeled as information resource "TerrainMap") holds information on whether any locations are open or have obstacles that the rover needs to maneuver around, see the lunar map in Figure 3.11. The terrain model used in this thesis is derived from a satellite photo of the west side of the Plato Crater, which is a location of interest for a human exploration mission for its geological features. The computational modes of the taskwork work in conjunction to update knowledge of the rover's surrounding and the feasible paths and waypoints to reach its destination. For example, the "LocalizeObstacles" action is modeled to check the surrounding area (i.e., get the "TerrainMap" resource) and update the knowledge of obstacles (i.e., set the "RockLocations"). Action "PlanRoverPath" produces a path from the rover's current location to desired location using the A* pathfinding algorithm (Hart, Nilsson, and Raphael, 1968). Finally, the "Select Next Waypoint" action provides the rover with the next desired waypoint to traverse to.

The computational work model can then be simulated to evaluate how the dependencies determine when, and how, the work can be executed. Figure 3.10 shows the timelines of when each action can be executed, together with plots of the rover's forward velocity for the first 20 minutes of the simulation. The taskwork actions are repeated when the rover reaches a waypoint, to re-evaluate the state of the work environment and modify states that are input to the rover dynamics. These iterations of the taskwork, amongst others, re-evaluate the obstacles around the rover, modify the planned path (accounting for any newly identified obstacle zones), and set the next waypoint to drive to. Once the sequence of actions is completed, the rover can start driving to its next waypoint.

Figure 3.11 shows the path of the rover obtained from the simulation results. The rover starts in the lower-left corner, with the goal of driving to the lower-right corner of the map. As the rover drives, the work model updates the waypoints based on new information. For example, initially the rover starts driving to the lower-right corner in a straight line, not aware of the crater that obstructs this path. At the first waypoint, when new rock locations are identified, the rover makes a sharp turn to the left to drive around the crater.

Thus, there is an intricate interaction between the work, the work environment, and the dynamics of the rover that can be evaluated with the simulation of computational work models. The simulation results presented in this case study can be extended to analyze how features of the work environment and characteristics of the work can drive the effectiveness of a team design. For example, the results could help identify where the work dynamics results in agents idling for extended periods of times, or, likewise, create spikes in taskload that might overload agents or create bottlenecks in the team's operations.



(a) Taskload limit of one.

(b) No taskload limit.

Figure 3.10: Timelines of taskwork and rover's forward velocity.

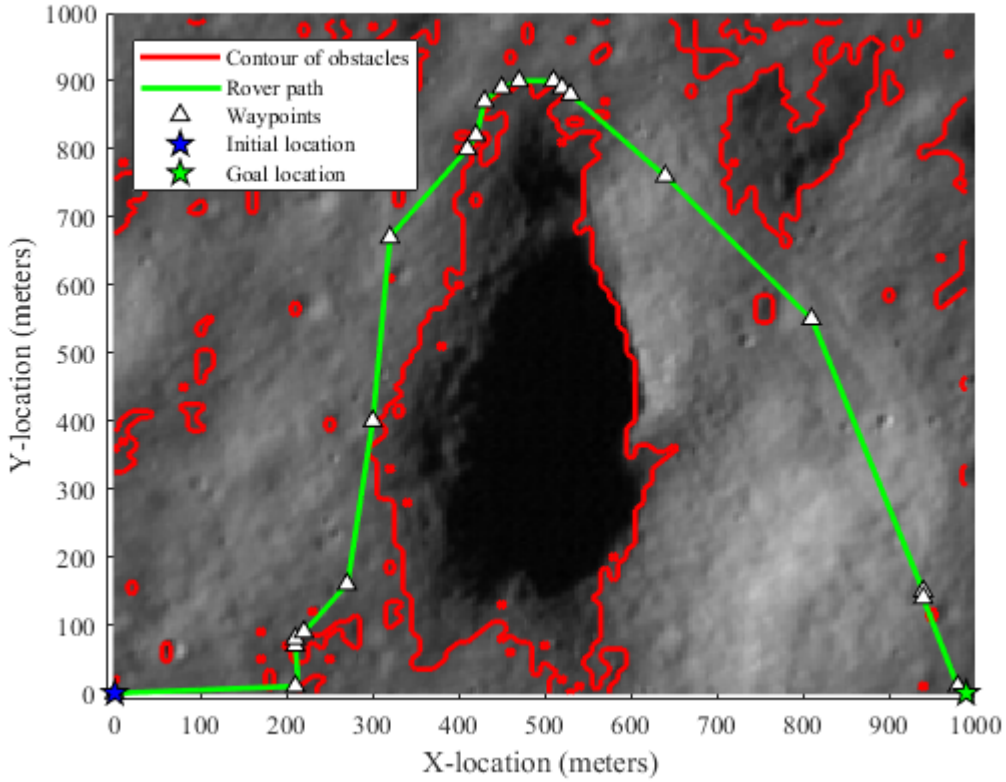


Figure 3.11: Path of rover.

3.6 Discussion

Analysis of work dynamics (through computational work models) can provide valuable insight in how constraint and dependencies in the work and work domain drive behavior in socio-technical systems. The explicit consideration of dependencies between a team's collective work and information and physical resources in the work environment can provide insight in how patterns originate, and how the work of individual and multiple agents is interweaved. Analysis of work dynamics can provide valuable insight in how these dependencies drive behavior in socio-technical systems.

Models can be refined iteratively, where simulation notably informs design of teams, but also the modeling itself. Indeed, the modeling of work and dependencies, and the subsequent simulation to evaluate the work dynamics, can be performed in an iterative manner. Even when just a notional idea of the work is available, designers of teams can start thinking through high-level dependencies that might govern the work. The conceptual models can then be refined by modeling dependencies in higher detail, identifying missing actions, resources, or dependencies. For example, where two actions occur in parallel in the simulation, but this is judged by the designer to be physically impossible, it is likely that a constraint is missing, such as a dependency with a physical resource or a locational constraint.

Likewise, models can be coarse at first, but then as the design progresses and one has more insight of the interesting aspects of the team and/or work model, the model can be refined to conduct more detailed analysis. Thus, as the design process progresses, the abstraction-decomposition space can be expanded into lower levels of decomposition, to evaluate work dynamics in a more fine-grained fashion.

To perform the type of dynamic analysis proposed here, a challenge is that designers need to have at least a notional idea of the typical durations of the actions. Estimates can be obtained from earlier data if available. If designing for a first-of-a-kind operational setting, however, estimates could initially be based on best-guesses and/or first principles of physics (e.g., when considering actions that involve physical processes such as locomotion). Then, once the ADS and associated work model have sufficiently crystalized, data can be obtained from simple human-in-the-loop evaluations of individual or collections of actions. To determine which actions to focus on in determining the durations,

one should consider how dependent each action is with others: for actions with low dependencies with the rest of the work model, correct estimates of the duration are less critical than for actions with high dependency.

Another possible challenge is that the modeling can be time consuming. The steps to setting up the work model (particularly creating the computational versions of the work) requires careful thought through the essence of each action. However, once the general work model is created it can be applied to many different scenarios and action sequences. Additionally, thinking through the dependencies is a useful formative undertaking for the designer.

Finally, in creating dynamic models of the work, there is a danger of getting into discrete event modeling which does not truly reflect the interactions between work and the work environment. There is a strong temptation to hard-code precedence relationships between actions when attempting to simulate work dynamics. The focus of the analysis, however, should be on how dependencies between work and the work environment, as represented by the physical and information resources, govern how the work can be interweaved. Explicit consideration of these dependencies provides formative insight in how team members must manage their work.

With discrete-event modeling the actions and dependencies are reduced to a normative model that prescribes a linear sequence of work. Generally, the analysis presented in this chapter can quickly become a normative one – specifying one way of doing the work in the team. Thus, there is instead a need for formative insight into how dependencies in the work drive the availability of different work patterns or sequences. The next chapter will

go into more detail on how a model of dependencies can be used to identify multiple ways that work can be executed.

3.7 Summary

This chapter discusses modeling and evaluation of work dynamics in teams. Work dynamics is defined as the processes by which dependencies in the work result in work patterns. As teams manage their work, these dependencies must be managed deliberately, as they constrain which actions can be performed when. In conceptual design phases of teams, there is a need for formative insight in how work dynamics evolve as a consequence of design decisions. This chapter therefore introduces a modeling and simulation approach that allows explicit evaluation of these dynamics.

To conduct such dynamic analysis, the work of a team is first modeled at various levels of abstraction. These levels describe the work from different perspective, from the highest-level purposes of the work to the tangible forms of artifacts and information. When relating these levels to each other designers can identify dependencies between the collective work of a team and the work environment as represented by physical and information resources. These dependencies drive the feasible work patterns.

Modeling of these dependencies then forms the basis for evaluation of work dynamics. Actions and resources, and the dependencies between them, can be modeled in computational form, and subsequently simulated to evaluate how these dependencies create patterns of work. A demonstration of this approach included a case study of on-orbit maintenance tasks on a spacecraft.

As discussed in the previous chapter, to design teams that can adapt to changing work demands, there is a need for systematic methods to conduct formative analysis of work strategies. The current chapter has outlined an approach to evaluate strategies, but not yet any systematic method for identifying feasible sequences from the ADS modeling of work. Thus, the next chapter will address this gap by further formalizing the model of work into a graph networks of a team's collective work and using these graph networks to identify multiple feasible action sequences.

CHAPTER 4. COMPUTATIONAL MODELS TO IDENTIFY FEASIBLE WORK STRATEGIES

Teams in complex work domains need to adapt to context. Studies of expert workers in such domains teach us that much of a system's resilience originates from its workers' ability to adapt their work strategies, to manage both performance and workload levels (Sperandio, 1978; Woods and Hollnagel, 2006). To support such adaptation, designers can build in flexibility to allow team members to "finish the design" (Vicente, 1999). Many current design methods for teams, however, inherently prescribe normative work strategies through implicit assumptions about how the work ought to be done.

Additionally, existing attempts to create a more formative approach to the design of the team (e.g. Ashoori and Burns, 2013) have applied static work models that cannot account for the evolving dynamics of the work itself, and the coordination and synchronization it requires within a team. Thus, to design teams that can adapt to changing work demands, models are needed to conduct a systematic analysis of work strategies.

This chapter introduces a systematic way of identifying feasible work strategies: the computational models of work discussed in the previous chapter are taken as a basis for an analysis of the network structure of actions, information resources and physical resources. Ultimately, this thesis covers both taskwork and teamwork, but this is broken down in two parts. This chapter identifies the first part – strategies driven by the dynamics of the taskwork. Work models are described as graph networks to provide formative insight in the constraints and dependencies that limit or drive adaptation in teams, whose dynamics

can then be analyzed in detail through computational simulation. To illustrate, this chapter includes two case studies that explore different work strategies.

4.1 Designing for Multiple Work Strategies

As discussed in chapter 2, to design teams that can adapt to changing work demands, models are needed to conduct a systematic analysis of work strategies in teams. Vicente (1999) defined strategies as the processes by which control tasks can be achieved. Whereas the definition of the task itself is a simple description of input and output, strategies define the possible ways by which a task's output can be realized. In the original work by Rasmussen (1981), a strategy is formally defined as "a category of cognitive task procedures that transforms an initial state of knowledge into a final state of knowledge." Rasmussen purposely defined strategy as a category, to contrast it with individual task procedures which are isolated, non-recurring instances. Context requires slightly different procedures in each situation, but, as Rasmussen argued, different specific instances of these procedures can be categorized together based on similar and stable characteristics (Rasmussen, 1981).

This dissertation focuses on reasoning about work in teams, where actions need to be coordinated between team members to together achieve higher-level goals. Thus, this work does not explicitly consider strategies that represent different internal processes or cognitive strategies of individual agents (e.g., information processing strategies, diagnosis strategies), but focuses different strategies for managing work within a team of multiple agents. To distinguish this type of strategy from others, it will be referred to as work strategy.

From a work strategy perspective, adaptation is selecting the work strategy that is appropriate to the immediate context. Work strategies can differ in a number of ways:

- What actions are performed, e.g., programming and engaging the autoflight system versus manually controlling the airplane.
- When actions are performed, i.e., differences in the ordering and timing of a fixed set of actions.
- How actions are performed, e.g., when supervising a process or team member, a worker can monitor different information, and at different times (Pritchett and Bhattacharyya, 2016).
- The actions' relations to the work goals, e.g., strategies that are aimed at structuring the work environment now to mitigate workload when work demands increase in the future versus strategies that are quick responses to immediate stimuli.

Chapter 2 discussed the contextual control models of cognition. Hollnagel (1993) formulated contextual control modes that describe different human behavior, or cognitive strategies, in response to contextual factors. Primary factors in this model include perceived available time and determination of outcome of previous actions. Secondary factors include the number of goals that are considered, availability of plans, mode of execution, and event horizon. Based on these factors, human workers operate in different modes of control, on a continuum between scrambled control (or panic mode) and strategic control (or optimizing mode).

It can be argued that the differences in work strategies described above are reflective of an individual’s contextual control. For example, two work strategies with differences in

their actions' relations to the work goals could be indicative of shorter or longer event horizons, reflecting the different time frames in which each of the work goals is managed. Likewise, work strategies with differences in what and how actions are performed could be reflective of the worker's perceived available time: short-duration work strategies with quick information sampling would be reflective of opportunistic modes of control, versus strategic control with more elaborate information sampling.

Effective adaptation, then, is based on the worker appropriately selecting work strategies to match the current demands, through the careful balancing of demands with the available resources, as reflected in different contextual control modes. Thus, each work strategy has different costs associated with it, in terms of the demands it poses to the agents and the resources that it requires. For example, effective strategies must match the perceived available time for the work, and the appropriate event horizon per the worker's control mode. This is true both for individual agents as well as across an entire team.

In this chapter it is argued that adaptation can be fostered by designing for multiple work strategies. To provide formative insight in a team's ability to adapt, this chapter discusses the analysis of flexibility in a team's work structure that allows for different work strategies, without making inferences on what is the "best" or optimal way of performing the work (as often the case in normative analysis such as scheduling). Thus, with flexibility identified, the aim is to maintain that flexibility through design, allowing the practitioners or workers to then "finish the design" in response to immediate context that they are experiencing during actual operations. As a basis for identifying flexibility, then, the question is: what are the constraints and how do these drive feasible action sequences?

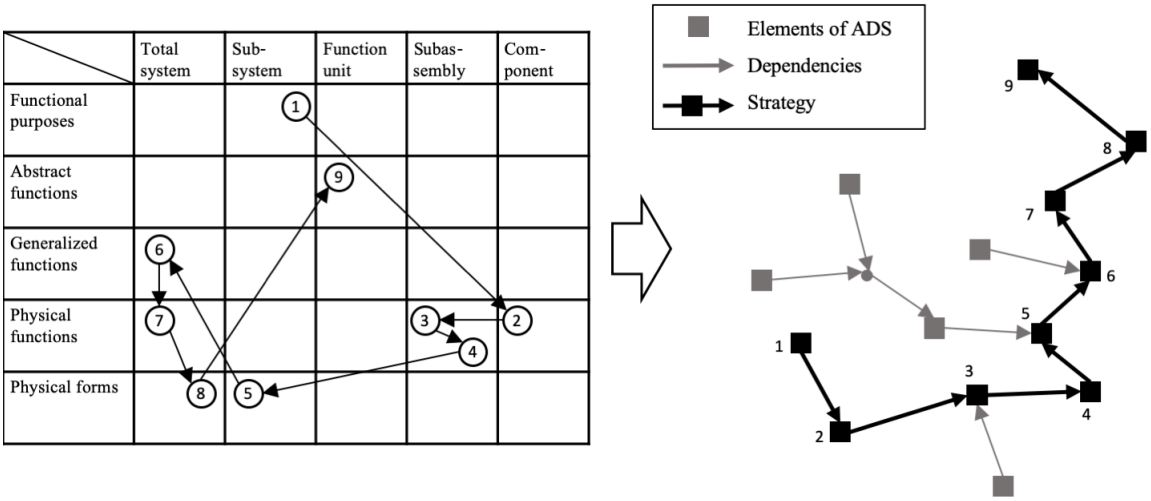


Figure 4.1: ADS with a work pattern, adapted from Rasmussen et al. (1994) (left) and a network representation of the ADS (right).

4.2 Network Representations of Work Models

In Rasmussen et al.'s original work on analysis of cognitive systems (Rasmussen et al., 1994), strategies are described in relation to the ADS. Rasmussen et al. argued that when agents perform work they move through the abstraction-decomposition space, jumping between multiple levels according to the means-end relationships. Thus, agents consider elements of the work at different levels of abstraction, from the highest-level goals to the lowest-level physical representation. This is illustrated in the left part of Figure 4.1, which is adapted from Rasmussen, Pejtersen, and Goodstein (1994).

Rasmussen et al. (1994) and later Vicente (1999) also argued that, when moving through the ADS, as an expert worker performs work on the work environment, he/she follows certain trajectories or patterns that abide by the means-end relationships between the different elements of the ADS. This is reflective of ecological psychology, and often

illustrated with the ants on the beach parable: the beach with its sand reliefs represents the properties and constraints of the work environment. The ants then traverse the beach in logical patterns driven by the beach's relief, e.g., following the trenches. Likewise, the relations in the ADS drive a human worker's strategies.

From this perspective, dependencies between the different levels of abstraction of the work domain determine what patterns of actions make sense, providing a basis for the identifying of feasible work strategies. Dependencies going up the ADS reflect how different parts of the work relate to priorities, values and the purpose of the team. Thus, these higher levels of abstraction are of a more intentional nature. When considering work strategies, these dependencies show how workers consider (and perhaps trade-off) the goals, values and priorities for the system. The next chapter discusses some of these higher-level considerations as summations of a work allocation in a team.

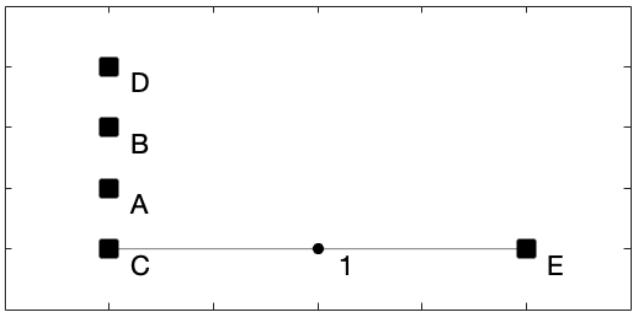
Dependencies going down the ADS reflect how physical representations of the work domain drive the feasibility of work strategies, as discussed in chapter 3. Thus, these dependencies between lower abstraction representations of the work environment and the work functions can provide formative insight for analyzing the flexibility in a work model and identifying multiple feasible work strategies.

This thesis proposes graph theory as a basis for a systematic analysis of dependencies and identification of feasible work strategies. Graph theory is the study of graphs, i.e., mathematical structures for modeling links or relations between objects. Through graph theory, an ADS can be represented as one or more graph networks, shown on the right of Figure 4.1. A graph consists of nodes that are linked through edges. Thus, nodes would

represent the elements of the ADS, and edges capture the dependencies between any two elements of the ADS.

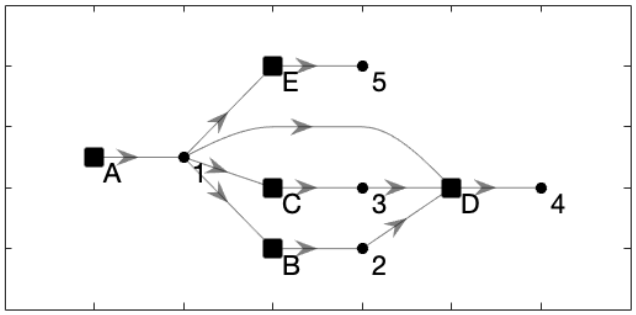
The level of decomposition along the horizontal parts-whole dimension can be tailored to the needs of the analysis being conducted. The one requirement is that any decomposition results in work descriptions that are elemental enough to be executed by individual agents, as will be relevant in the next chapter on work allocation in teams. Any judgment on the desired level of decomposition is dependent on what is deemed most relevant for the analysis. For example, particularly dynamic tasks could be modeled in more detail to the right, whereas as others are left as coarser descriptions. Furthermore, throughout a design process, it would make sense to start at a fairly coarse level of decomposition, to later move to finer-grained decompositions when the design has further crystalized.

Two different types of graphs capture two different types of dependencies that determine the dynamics of the taskwork. The first graph of the work model denotes the dependencies between actions and physical resources. This is a non-directional graph that shows what physical resources are required and therefore occupied by an action through its execution. In a non-directional graph, the edges are symmetric (when A is linked to B, B is also linked to A). Thus, any actions linked to the same physical resource cannot be executed in parallel. Figure 4.2 shows an example with five actions (A-E) and one physical resource (PR1). The linkages here indicate that both action C and E need the use physical resource PR1. The other actions do not require a physical resource for their execution.



Actions	Resources
A.ActionA	1.PR1
B.ActionB	
C.ActionC	
D.ActionD	
E.ActionE	

Figure 4.2: Example of non-directional graph of actions and physical resources.



Actions	Resources
A.ActionA	1.IR1
B.ActionB	2.IR2
C.ActionC	3.IR3
D.ActionD	4.IR4
E.ActionE	5.IR5

Figure 4.3: Example of directional graph of actions and information resources.

The second graph captures the dependencies between actions and information resources. Figure 4.3 shows an example with five actions (A-E) and five information resources (1-5). This is a directional graph that shows action's dependencies with information or states of the work environment. A directional graph has asymmetric edges, represented as arrows. Thus, this directional graph represents how information resources are shared between actions, wherein edges from information resource to action represent that action requiring the information, and, likewise, edges from action to information resource represents that an action provides that information, during or following the completion of the action.

Additionally, a distinction can be made between standalone information resources and information resources that are attributes of physical resources. Some of the information resources are intangible, elemental pieces of information, others are attributes of, and therefore connected to, a physical resource. There might be a physical action required to satisfy any dependencies for this latter type of resources: when information resources are part of a physical resource, it implies there is a requirement for physical interaction. For some attributes, this in turn implies that the physical resource that is associated with this information resource needs to be in the same location as the agent.

These two graphs together capture the sharing of physical resources and information between actions that drive the available set of work strategies. When considering the ADS in network form, the relationships between the levels can be analyzed in a systematic way. The network representation can be used to perform a broad analysis of the work model, as any work strategies need to abide by the structure of the work and the work environment as reflected in these graphs. Some constructs from graph theory that can be used to analyze the ADS include clusters, which can help reveal natural groupings and clusters in the work domain (the next chapter describes this in more detail); the density, which could reveal how connected and structured a work domain is; and the degree and centrality of nodes, which could be analyzed to identify the importance and/or effect of a particular element of the ADS to the broader work domain. The next section describes how these network representations can be used to systematically identify feasible work strategies.

4.3 Identifying Feasible Work Strategies

This section uses the graph representations of the actions and resources to characterize the flexibility in a team's collective taskwork. The network of actions and information resources provides insight into sequences of actions that transform the work environment in structured ways. The network of actions and physical resources provides additional insight on what actions cannot be performed in parallel due to physical and/or spatial constraints, where certain actions need to take place at particular locations.

4.3.1 Multiple Paths

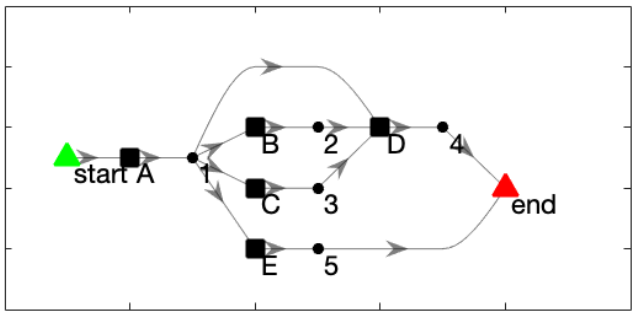
First, only some edges of the graphs are relevant for the analysis of work strategies. Relevancy here depends on the purpose of the analysis, in terms of what transformation of the work environment is of interest, and how this is reflected in the graphs. For example, analyzing a large work model of air traffic management operations for work strategies for conflict resolution, only part of the graph is needed. One way of identifying the necessary nodes is to identify the tree created by forward propagation from the initial state, choose a goal node (information resource) representing the outcome of interest to this strategy, and identify the tree created by backward propagation from this node. Then, the intersection of both trees can be selected as the subgraph for further analysis. Furthermore, for the remainder of the analysis proposed here, graphs must be acyclic. If the graph of actions and information resources contains cycles, these must be removed from the graph.

Then, to identify the set of feasible work strategies, the current state of the work environment is defined. With an acyclic graph of actions and information resources, a virtual "start" node can be connected to all action and resource nodes that have "indegree"

of zero (i.e., these actions do not get any information resources, and these information resources are not set by any action). Similarly, a virtual "end" node can be connected to all action and resource nodes that have "outdegree" of zero (i.e., these actions do not set any resources, or these resources are not gotten by any follow-up action). Continuing the example discussed in the previous section, the graph with virtual start and end nodes is shown in Figure 4.4.

The linkages represent what can be done, but not necessarily what needs to be done. Thus, between the initial and goal states, agents may take different paths representing multiple feasible work strategies. One can carefully consider the different dependencies and how they might be used for different work strategies. Throughout the analysis, certain edges in the graph can be made active or inactive, based on different paths that are identified.

This process identifies work strategies that differ in what actions are performed, and how much and what information each action is based on. For example, there might be shortcuts that can be taken, based on two paths with different actions leading to the same



Actions	Resources
A.ActionA	1.IR1
B.ActionB	2.IR2
C.ActionC	3.IR3
D.ActionD	4.IR4
E.ActionE	5.IR5

Figure 4.4: Graph with virtual start and end nodes.

state. Or, different dependencies representing different information requirements can be considered. For example, when quality of information improves, certain dependencies might become irrelevant. Likewise, the analysis can identify different degrees of anticipating future information (not including a dependency) versus waiting for information to become available (including a dependency). Such considerations are especially important in the discussion in the next chapter, on interaction modes for coordinating work between multiple agents.

Graph theory has a variety of path finding algorithms that can facilitate the analysis of different paths. A wide range of such methods is available with different purposes. For example, Dijkstra's algorithm (Dijkstra, 1959) is commonly used to identify the shortest path between two nodes of a graph, or to create a shortest-path tree from an initial node to all other nodes of a network. However, it should be noted that the goal of the analysis here is not to identify only the shortest path, but the breadth of feasible paths, not excluding longer paths which also represent feasible work strategies.

Additionally, once multiple paths are identified, the graph of physical resources and actions poses additional constraints in terms of what actions can be executed in parallel and which need to be performed in sequence. These dependencies are undirected and can therefore be realized in two ways: first action 1, then action 2, or vice versa. The combinatorics could blow up very quickly when multiple resources are used by multiple actions, although some of them might turn out to be infeasible based on the information resources network (thus, checks can be implemented to assess feasibility before creating an entirely new option).

Other implications may be associated with physical resources linkages, such as requirements for fetching and traversal (as will be discussed in case study I). For example, the locations of resources and agents could be represented as information resources (e.g., an attribute of a tool). The network of physical resources and actions can then help identify where there is a need to directionally link taskwork actions with fetch and traversal actions. These additional requirements are particularly relevant in a multi-agent scenario in which the tools need to be shared between multiple agents doing simultaneous work. For example, the geospatial configuration of the team and the work requires agents to traverse and/or fetch physical resources, in the worst case at opposite ends of the work space.

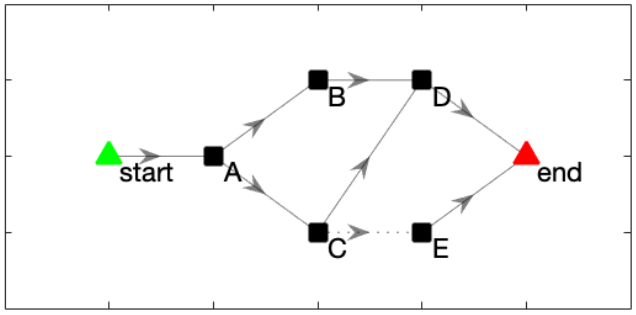
To account for the precedence constraints created by the physical resources, these can be added to the graph of actions and information resources as edges between two actions that use the same physical resource. Provided that no dependencies with information resources are violated, these edges can be added in two directions as discussed earlier. Thus, different precedence constraints can be added resulting in different versions of the lower-abstraction information resource network that can then be used for further analysis. Finally, the graphs of information resources and the graph of physical resources can be merged into multiple versions of graphs consisting solely of actions, representing the inherent order of actions that abides by both the information as well as the physical resource dependencies.

Going back to the example, because C and E need to use the same physical resource, two work strategies are possible: first action C, then E, or vice versa. To obtain a sequence of action (without the resources), the resource nodes can be substituted with directional edges between actions. In addition, redundant dependencies can be removed through a

transitive reduction on the graph. The graph for the two work strategies is shown in Figure 4.5. Solid edges represent dependencies of information resources; dashed edges represent dependencies of physical resources.

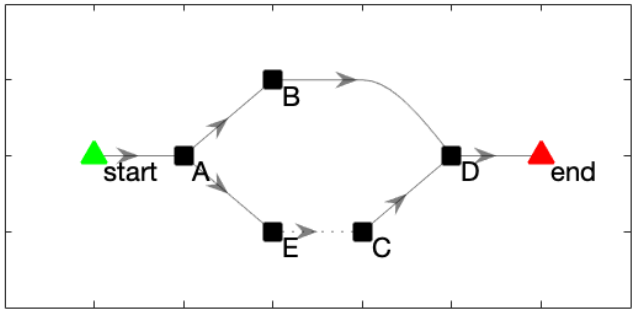
4.3.2 Topological Sorting

Within a single work strategy, multiple paths between the start and end nodes represent multiple sequences that, based on the work characteristics, can occur in parallel. However,



Actions
A.ActionA
B.ActionB
C.ActionC
D.ActionD
E.ActionE

(a) Work strategy 1



Actions
A.ActionA
B.ActionB
C.ActionC
D.ActionD
E.ActionE

(b) Work strategy 2

Figure 4.5: Two work strategies, accounting for both information and physical resource dependencies.

the taskload limit of agents can be a constraining factor in the execution of parallel actions. Any parallel threads of actions can be interweaved in different ways, which an agent or a team of agents would need to do when these actions cannot be executed in parallel due to taskload constraints. This is relevant for an individual performing the work, but also in a team, where the specific requirements for interweaving the work is influenced by the work allocation.

The analysis so far has not considered in what order actions are performed. The first phase of the analysis has answered the question of what actions are performed for different work strategies, as a result of how actions are performed, and what and how much information each action is based on (as reflected in their dependencies with other actions). The second phase then, discussed in this subsection, considers the different orders in which these actions can be executed.

In graph theory, the different ways of interweaving actions can be found through computing topological sorts. Thus, once the appropriate set of actions and dependencies with resources have been identified in the first phase of this analysis, the feasible topological sorts can be identified. In a topological sort, all directional linkages are forward in time. Thus, in a topological order of actions, information resources are set by an action before they are gotten by another and physical resources are not used by the two actions at the same time.

These topological sorts can be performed between the start and end nodes. Varol and Rotem (1981) created an algorithm that can compute all possible topological sorts. The problem of calculating all topological sorts is, however, NP-complete. Thus, with relatively

small work models the computational time is manageable, but for larger problem spaces it is necessary to find alternate ways of analyzing topological sorts, such as looking for specific sorts with knowingly different characteristics. One such way would be to analyze differences between topological sorts found through depth-first versus breadth-first search algorithms.

In the earlier example, eight different topological sorts are feasible. For work strategy 1, the feasible sorts are A-B-C-D-E, A-B-C-E-D, A-C-B-D-E, A-C-B-E-D, A-C-E-B-D. For work strategy 2, the feasible sorts are A-B-E-C-D, A-E-B-C-D, A-E-C-B-D.

4.3.3 Simulation

The different work strategies can be simulated to explicitly account for the temporal dynamics of the work. By considering the dependencies between the work and resources directly in the identification of feasible work strategies, these temporal effects can be evaluated without many modifications to the work modeling: Both the graph analysis as well as the computational models of work use the same elements and dependencies between them. Thus, once a computational work model is established, the identified work strategies are set as input parameters to the simulation, allowing for fast-time evaluation of many different work strategies.

As an input parameter, the work strategies are used as templates that are modified by the simulation as it evaluates the dynamic effects of the dependencies. The simulation can provide additional insight beyond the static analysis of the graph by assessing the specific timing of actions based on evaluating the interaction between the work and the work environment (and any dynamic processes that might be ongoing in the work environment).

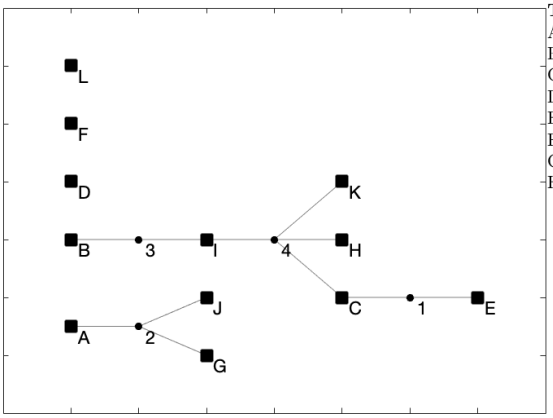
For example, the locomotion of agents or vehicles, orbital mechanics, or aircraft dynamics could influence the appropriate timing of actions. Such dynamical processes may change the state of the information and physical resources, thereby rendering certain work strategies infeasible or requiring deviation from the template to accommodate the changing work environment.

Beyond evaluating the work dynamics in terms of the timing of activities, simulation may also evaluate the appropriateness of the work strategies with different contextual factors. Conditions of the work environment can be simulated through the values of resources, allowing the analysis of how different environmental conditions would render certain work strategies more or less effective. In space operations, for example, different communication delays can be simulated to test how those might affect the feasible set of work strategies and a team's ability to adapt.

Thus, through simulation the generated dataset of feasible work strategies can be analyzed in more detail, providing many opportunities for evaluating work strategies in different contexts. Of note here is that the simulation results might show common characteristics of work strategies, which can then be used to group together strategies, reflecting Rasmussen's (1981) definition of strategies as categories of task procedures. For example, strategies could be grouped based on how proactively or reactively their actions manipulate information or physical resources, reflective of similar look-ahead horizon.

4.4 Case Study I: Maintenance

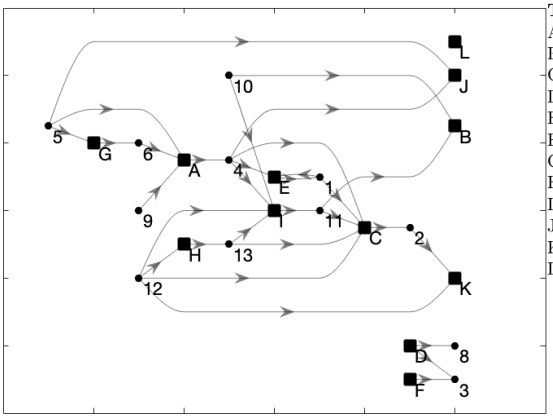
The case studies presented here will demonstrate how network visualizations can provide formative insight to designer, through analyzing constraints in the work and identifying



Taskwork
A.ApplyInspectionTools
B.DisposeOfBrokenPanel
C.EmplaceNewPanel
D.EnterDock
E.GetNewPanel
F.LeaveDock
G.PrepareInspectionTools
H.PrepareRepairTools
I.RemoveBrokenPanel
J.StoreInspectionTools
K.StoreRepairTools
L.Traverse

Resources
1.BackupPanel1
2.InspectionTool
3.Panel1
4.RepairTool

Figure 4.6: Network representation of the actions and physical resources for the on-orbit maintenance scenario.



Taskwork
A.ApplyInspectionTools
B.DisposeOfBrokenPanel
C.EmplaceNewPanel
D.EnterDock
E.GetNewPanel
F.LeaveDock
G.PrepareInspectionTools
H.PrepareRepairTools
I.RemoveBrokenPanel
J.StoreInspectionTools
K.StoreRepairTools
L.Traverse

Resources
1.BackupPanel1Location
2.BackupPanel1ScrewedOn
3.DockAvailable
4.InspectionResult
5.InspectionToolLocation
6.InspectionToolStatus
8.MissionComplete
9.Panel1Condition
10.Panel1Location
11.Panel1ScrewedOn
12.RepairToolLocation
13.RepairToolStatus

Figure 4.7: Network representation of the actions and information resources for the on-orbit maintenance scenario.

feasible work strategies. First, the work model from chapter 3 is represented as two graphs: one for the actions and physical resources, see Figure 4.6, and one for the actions and information resources, see Figure 4.7. In the physical resources graph, inspection actions (A, G, J) are dependent through the inspection tool, and the actions for replacement of the panel are dependent through three physical resources: the backup panel (1), the panel (3) and the repair tool (4). In the information resources graph, most actions are highly interconnected, with a few exceptions: Traverse (L), EnterDock (D) and LeaveDock (F) are independent.

There is only one order of actions in which the physical resources can be shared; the order of actions is also constrained by the dependencies due to information resources. Figure 4.8 shows the feasible work strategy, not accounting for required fetching operations.

The use of physical resources implicitly establishes that actions are dependent on the location of resources. These locations can be set by fetch actions; thus, there are

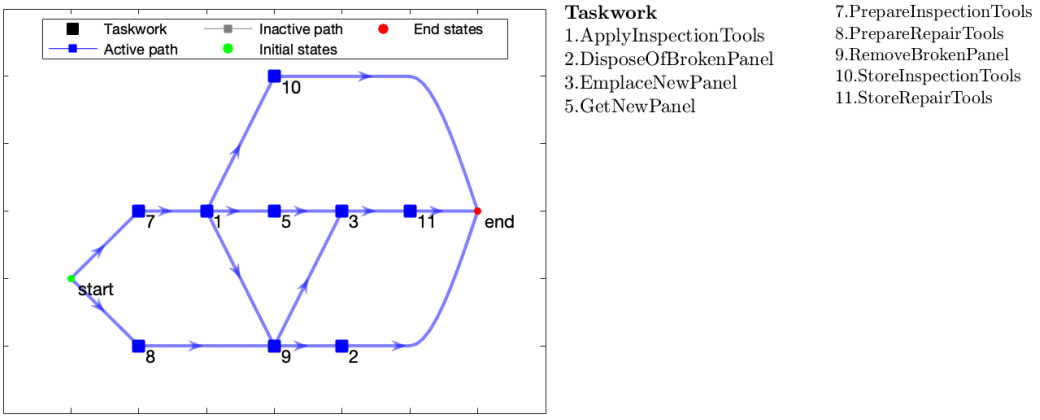


Figure 4.8: One feasible work strategy before linking fetching and traversal.

dependencies between the maintenance-specific taskwork and fetching operations. This case study considers different strategies for fetching the required physical resources. For example, agents have flexibility in how many physical resources they carry with them during a fetching operation (e.g., depending on the size of toolbelts, availability of carrying equipment, or the size of a physical resource, an agent might decide to carry one or more physical resources at once), and which physical resources they carry during each fetching operation (e.g., depending on context, physical resources can be fetched at different times and in different combinations). To identify and analyze these different work strategies, the network of information resources and actions with the network of physical resources and actions are combined, i.e., the physical constraints are translated into dependencies in the lower-abstraction information resources.

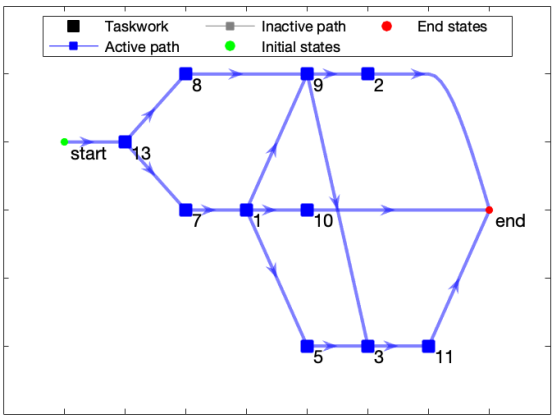
The initial state is assumed to have all agents, required tools and the backup panel (represented as physical resources) at the dock. Then, to model an action's dependency with the location of physical resources (where the locations are represented as information resources), when an action is linked to a physical resource, an additional dependency is added in the information graph between the action and the resource location, which can be seen in Figure 4.7. For example, prepare inspection tools (G) is linked to inspection tool location (5). There are three of these resources that have an explicit location requirement to them: the inspection tools, the repair tools and the backup panel. The fourth physical resource, the panel, is already at its correct location (as part of the spacecraft's exterior).

These location resource now need to be set by fetch actions, but these can be linked to the location resources in different ways. For example, for a single fetching operation carrying all three physical resources at once, the fetch action is linked to set all three

location information resources. Likewise, for multiple fetching operations, each fetch action is linked to its respective location resources. When fetching two physical resources, there are three ways in which the physical resources can be divided over the two fetching operations; one of these was picked for this demonstration, which is one fetching operation for the inspection and repair tools, and one for the backup panel.

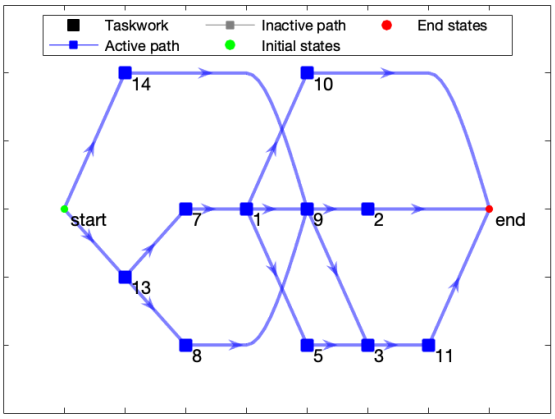
Thus, there are three feasible work strategies, identified as three different paths through the graphs, shown in Figure 4.9. The work strategy with three separate fetching operations (Figure 4.9c) has the most opportunity for parallel processes, which, depending on the work allocation, could be beneficial in a multi-agent team. Further exploration of these strategies could also consider how the fetching operation is dependent on inspection results: there are, for example, different work strategies in what information the fetching of the backup panel and repair tools is based on; these could be fetched right from the start, in anticipation of a required panel replacement, or could be dependent on the inspection result (4) in a more conservative and reactive work strategy.

Likewise, the work in each of these strategies can be executed in different orders. Certain orders of actions could be reflective of different contextual control modes, i.e., being more proactive or reactive in the timing of actions. For example, agents have flexibility in when required tools are fetched: an agent could first fetch and prepare all tools before starting inspection or wait until immediately before the tools are required. Thus, a further analysis of work strategies can compute all possible topological sorts of the directional graph.

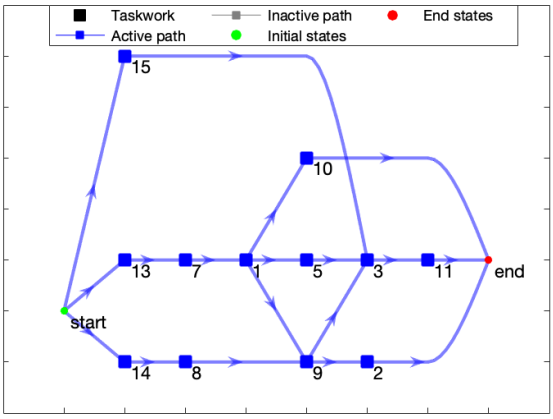


Taskwork
1.ApplyInspectionTools
2.DisposeOfBrokenPanel
3.EmplaceNewPanel
5.GetNewPanel
7.PrepareInspectionTools
8.PrepareRepairTools
8.PrepareRepairTools
9.RemoveBrokenPanel
10.StoreInspectionTools
11.StoreRepairTools
13.Fetch1
14.Fetch2
15.Fetch3

(a) Fetching in one operation.



(b) Fetching in two operations.



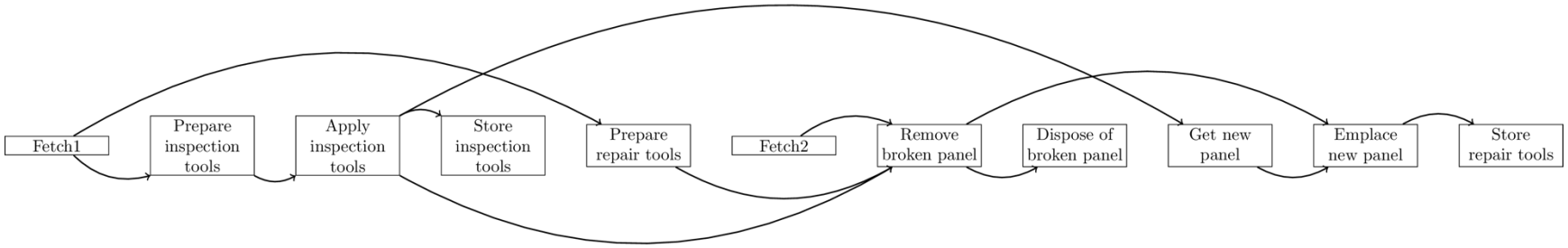
(c) Fetching in three operations.

Figure 4.9: Three different work strategies.

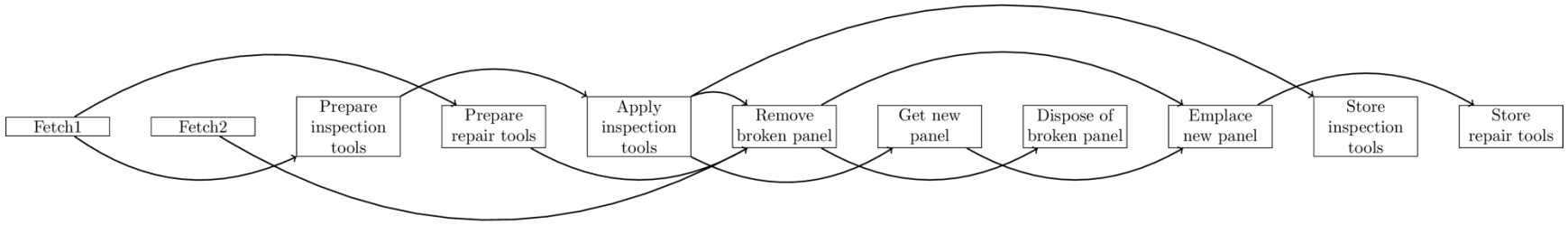
Using the algorithm developed by Varol and Rotem (1981), a total of 896 different topological sorts were identified for the work strategy with two fetch operations. Then, in post-processing, these sorts can be categorized based on similar characteristics. One way of categorizing the sorts is based on how many actions the dependencies span within each sort. This span of dependencies can be considered a measure for the event horizon. Figure 4.10a shows a topological sort with the maximum number of one-step dependencies. This would be an indication of a more reactive cognitive strategy, in which actions are chosen based on an immediate need for information resources, as a stimulus-response mechanism without explicit consideration of future steps, reflective of an opportunistic control mode.

Figure 4.10b shows a topological sort with the minimum number of one-step dependencies. In this case, with dependencies spanning multiple actions into the future, the work strategy is representative of contextual control that anticipates required states (as information resources) several steps into the future, reflective of a strategic control mode.

Finally, these work strategies can be simulated to assess the temporal dynamics of each work strategy. With the simulation's explicit modeling of the states of the work environment and the work itself, a wide variety of analyses of the work strategies can be performed. For example, one aspect that has not yet been accounted for in the static analysis is the required traversal of agents, in those cases where actions need to take place at different locations. Simulation can account for the location of agents and actions through time and automatically account for traversal time. More detailed assessment could also assess different strategies with multiple inspection outcomes, where some work strategies might turn out to be too proactive when a panel replacement is no longer needed.



(a) Topological sort with the most dependencies spanning one step.



(b) Topological sort with the least dependencies spanning one step.

Figure 4.10: Two topological sorts of work strategies with two fetch operations.

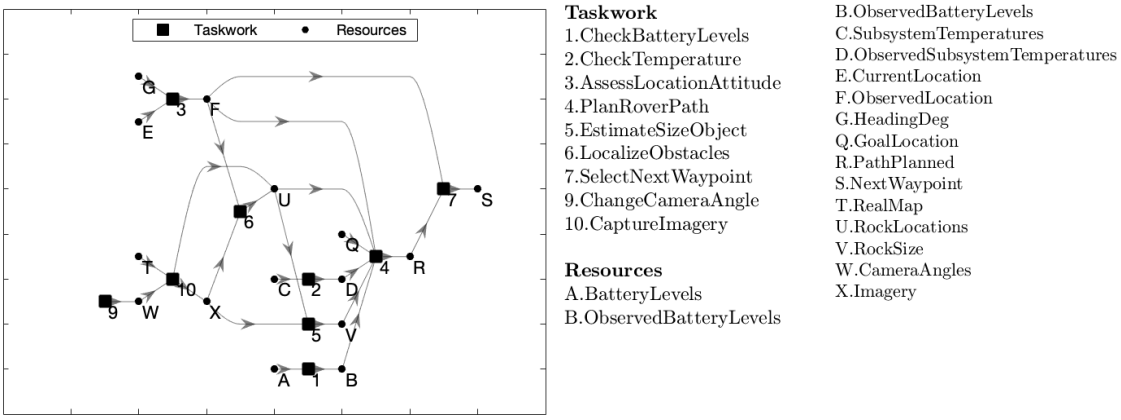


Figure 4.11: Network visualization of precedence relationships.

4.5 Case Study II: Lunar Rover

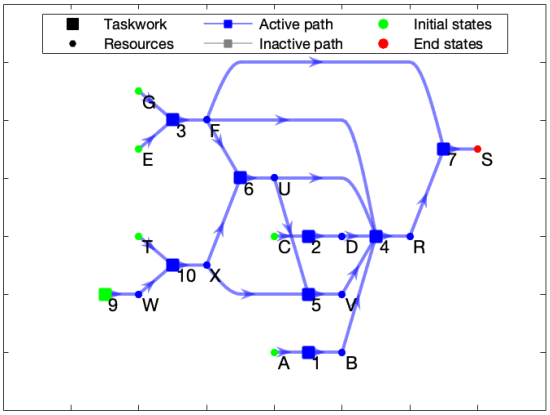
The second case study will further demonstrate how network visualizations can provide formative insight to designers through analyzing constraints in the work and identifying feasible work strategies. It continues the case study from chapter 3, modeling of the work of a team consisting of a rover and two astronaut drivers on a planetary surface.

Figure 4.11 shows the work model from chapter 3 in a graph representation. The nodes are actions (squares) and information resources (circles). The edges represent set and get linkages. The directional graph clearly shows the centrality of the path planning ("PlanRoverPath"). The element representing the rover dynamics in the ADS closes the loop from the information resource "NextWaypoint" to information resources that represent the rover states (e.g., "SubsystemTemperatures", "CurrentLocation", and "HeadingDeg"), but is not explicitly included in the graph to create an acyclical graph for identification of work strategies. The rover dynamics are, however, accounted for in simulation of the work dynamics, including its interaction with the taskwork in this graph,

through the information resources for the rover states. Throughout a design process, this model can be further refined iteratively by identifying new information requirements for actions and thereby creating new edges.

Once the network is fully fleshed out, it can be used to identify feasible work strategies between two or more points. Figure 4.12 shows three feasible paths through the information resources graph. Figure 4.12a is the full graph, accounting for all information resources and actions. The path in Figure 4.12b differs in that path planning considers only the rover's location, and the location and size of obstacles, without explicitly accounting for the subsystem temperature and battery levels. Thus, the branches of the graph associated with these information resources (i.e., checking these rover states) are not part of this path. Likewise, Figure 4.12c shows a path in which the subgraph for updating the information resources "RockLocations" and "RockSize" is not considered. Thus, although this information is still input to "PlanRoverPath", the actions for updating this information are not part of this path. Using this graph representation, other paths can be identified with similar reasoning about the dependencies between information resources and the actions.

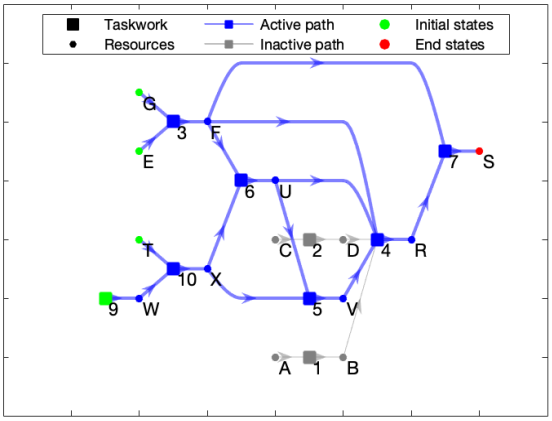
Each of the paths in Figure 4.12 represents a different work strategy, as shown in Figure 4.13. Work strategy 1 (Figure 4.13a), obtained from the full path, steps through all the taskwork actions, localizing obstacles and estimating their size before doing path planning, as well as accounting for the subsystem temperature and battery levels in path planning. In work strategy 2 (Figure 4.13b), the path planning does not account for the most recent observations of the battery levels and subsystem temperatures. Likewise, in work strategy 3 (Figure 4.13c) the rock locations and sizes are not updated, instead the path planning is performed without formal analysis of obstacles.



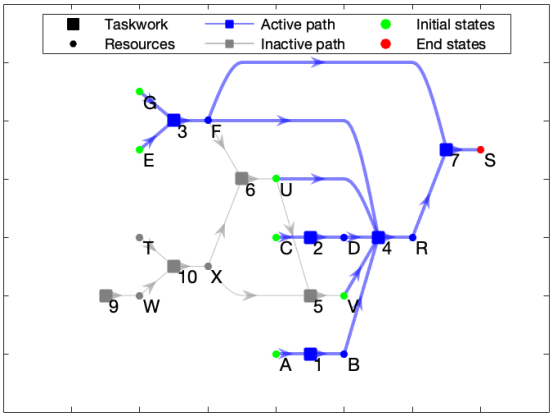
Taskwork
1.CheckBatteryLevels
2.CheckTemperature
3.AssessLocationAttitude
4.PlanRoverPath
5.EstimateSizeObject
6.LocalizeObstacles
7.SelectNextWaypoint
9.ChangeCameraAngle
10.CaptureImagery

Resources
A.BatteryLevels
B.ObservedBatteryLevels
C.SubsystemTemperatures
D.ObservedSubsystemTemperatures
E.CurrentLocation
F.ObservedLocation
G.HeadingDeg
R.PathPlanned
S.NextWaypoint
T.RealMap
U.RockLocations
V.RockSize
W.CameraAngles
X.Imagery

(a) Full path

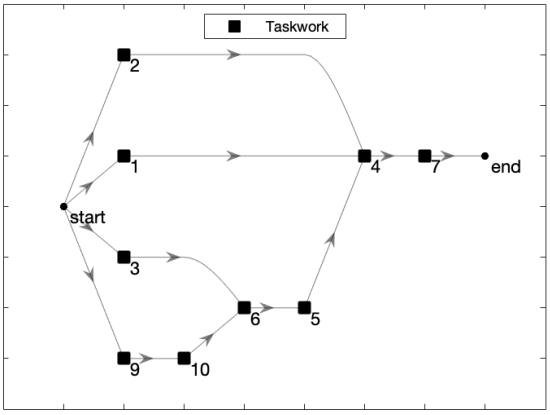


(b) Path with partial rover state checking



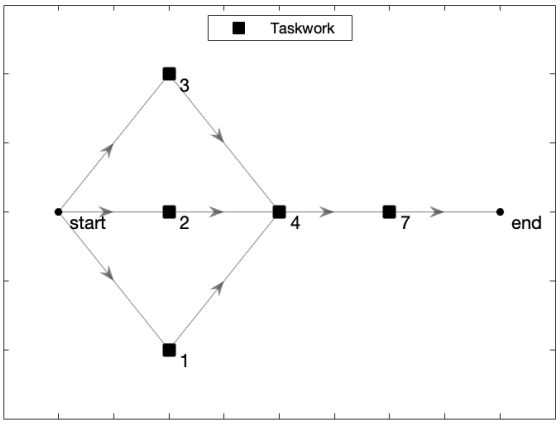
(c) Path with no obstacle detection

Figure 4.12: Three different paths through the graph of actions and information resources.

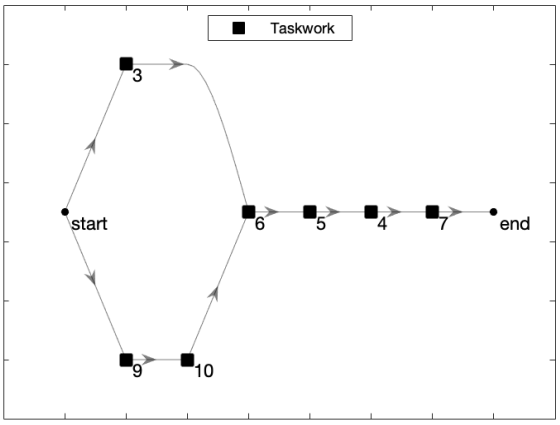


Taskwork
1.CheckBatteryLevels
2.CheckTemperature
3.AssessLocationAttitude
4.PlanRoverPath
5.EstimateSizeObject
6.LocalizeObstacles
7.SelectNextWaypoint
9.ChangeCameraAngle
10.CaptureImagery

(a) Work strategy 1



(b) Work strategy 2



(c) Work strategy 3

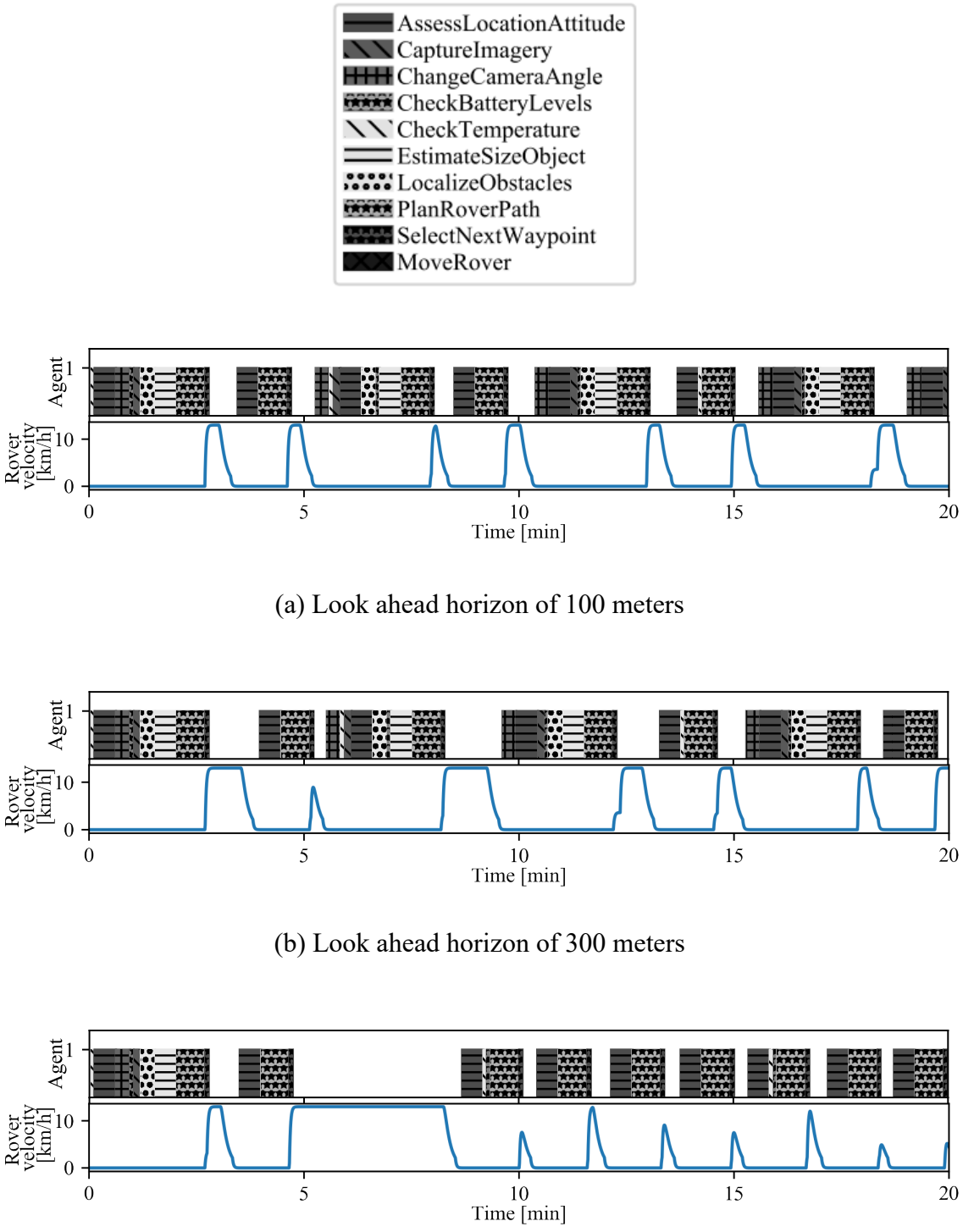
Figure 4.13: Three work strategies.

These three strategies are each appropriate in different contexts. One such contextual factor includes the quality of the information that is available to the agents, captured in the values of the information resources. For example, work strategy 1 would be appropriate when the quality of information for "rock locations" is low, and therefore the imagery needs to be frequently sampled and used to inform path planning. In contrast, work strategy 3 is appropriate when high quality information on the "rock locations" is already available and does not require constant sampling. In many potential missions, the information quality can vary with location and terrain, and thus the team may need to adapt between these strategies.

With these two work strategies identified, more detailed evaluation can be performed through simulation of the work. Through simulation, the cost and benefits of each work strategy can be evaluated through an explicit evaluation of how each work strategy interacts with the work environment. To illustrate each strategy's appropriateness in different context, the actions and information resources can be modeled to capture different degrees of information quality on obstacle locations and sizes, representing different contexts in which the work is conducted (e.g., terrain, location, atmospheric conditions, sensory capabilities could all affect the available information on obstacles). Specifically, different look-ahead times can be modeled for capturing imagery and updating the obstacle locations used for path planning. Thus, with short look-ahead horizons, lower-quality information is available for path planning, requiring the agents to conduct frequent imaging and obstacle detection, whereas with longer look-ahead horizons, it may be more appropriate to let the rover drive without re-evaluating obstacles.

Adaptation of the work strategies in response to information quality can be evaluated by having the simulation engine make local changes to the action sequences. Dependencies between actions and information resources can be characterized through quality-of-service (QOS) requirements, defining the minimum time frame during which information is considered accurate enough for conducting work. Thus, when an action is dependent on an information resource that does not meet QOS standards (e.g., due to work dynamics delaying actions beyond the QOS window), the information must first be updated through the appropriate actions. This would render certain work strategies infeasible and would require adaptation to use work strategies that fit QOS requirements and/or are more lenient in their QOS requirements. Thus, based on the QOS requirements at each iteration of the work model, the simulation selects the appropriate work strategy.

Figure 4.14 shows the timelines of when each action is executed as detailed by the simulation for different look-ahead times for capturing imagery. In this case, all actions were performed by a single agent to illustrate the dynamics inherent to the work even before accounting for inter-agent coordination. A look-ahead horizon of 100 meters (Figure 4.14a) requires frequent information sampling at the expense of stopping the vehicle, with these work demands requiring the agents to rely heavily on work strategy 1. However, when waypoints are spaced particularly close together (e.g., between 8 and 10 minutes), the obstacle locations do not need to be resampled, thus resulting in work strategy 2 being the more appropriate work strategy. Likewise, a look-ahead horizon of 300 meters (Figure 4.14b) allows waypoints to be placed further apart, requiring fewer re-evaluations of obstacles and the path planning. Finally, if all information on obstacles is known beforehand (Figure 4.14c) the rover can drive for extended periods of time without needing



(a) Look ahead horizon of 100 meters

(b) Look ahead horizon of 300 meters

(c) Complete map known beforehand

Figure 4.14: Timelines of actions with work strategy 1 (top) and work strategy 2 (bottom)

updated information. Thus, only one iteration of work strategy 1 is required to plan the path, but then the agents can update the waypoints through work strategy 2.

Correspondingly, Figure 4.15 shows the path that the rover traversed for each look-ahead time. Clearly, having prior knowledge of the obstacles resulted in a more optimized path. With lower information quality (look-ahead horizons of 10 and 20), the waypoints are spaced close together, to avoid the rover driving into an area with unknown obstacles. In addition, in these runs the rover first attempts to drive straight to the goal position, only later finding out (through the LocalizeObstacles and EstimateSizeObject actions) that this route is not a viable option. Several iterations of the work model are then needed to find a path around the obstacles.

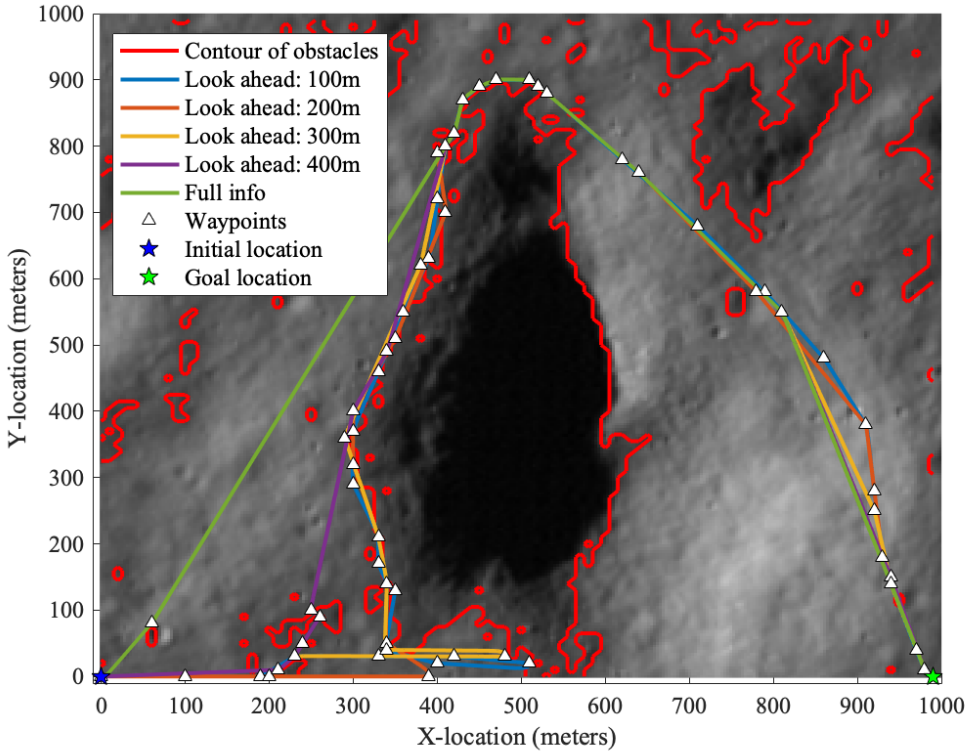


Figure 4.15: Rover paths for different look-ahead horizons.

The simulation can also examine a range of potential work allocations, as will be discussed in the next chapter. For example, an astronaut might plan the paths based on an assumed database of terrain information and the rover then move through them autonomously (correspondingly closely with Strategy 1), or the human may actively steer the camera to confirm the feasibility of the planned path and interact more closely with the rover in path following (as may be necessary with Strategy 2).

4.6 Discussion

How a team adapts is, as hypothesized, determined, first, by how many work strategies are available and, second, how well the attributes of feasible work strategies match the work demands of the team. Thus, effective design of teams should consider a range of feasible work strategies. To facilitate such a design process, the graph analysis of a work model presented in this chapter can provide formative insight into how characteristics of the work and the work environment can shape a team's behavior and ability to adapt. In addition, with methods from graph theory, graphs of actions and resources can be used to systematically identify the set of feasible work strategies. Moreover, the static analysis of graphs can then be used as input to a dynamic evaluation of feasible work strategies, explicitly accounting for the work dynamics as discussed in the previous chapter.

The two case studies demonstrate how graph representations of a work model can be used to analyzing flexibility and identify different work strategies in a team's collective work. The first case study showed that through changing the dependencies between actions, different work strategies can be identified. Additionally, the work strategies can be ordered in different ways, reflecting contextual control modes of Hollnagel (1993). The second

case study demonstrated the identification of feasible work strategies based on different information requirements, each leading to a different path through the information resource graph. This case study then also demonstrated how the work strategies can be simulated in a computational work model, and how each work strategy is appropriate in different contexts.

The analysis of graphs presented in this chapter can be performed iteratively, further refining the ADS of the team's work. While analyzing the different work strategies, missing elements of the ADS or missing dependencies between the levels of abstraction can be identified, which in turn could lead to a different set of feasible work strategies. In other cases, it might be relevant to change the level of decomposition to model certain phenomena in more or less detail. Such iterative design thus is also a form of formative insight, wherein an analysis of work strategies could help identify what elements of the ADS governs the work of a team.

Challenges with this approach include the computational complexity and large amounts of data that can be created when identifying the work strategies. Thus, for large work models, the network analysis needs to be mindful of what aspects of work strategies are relevant. Algorithms can identify the extremes in terms of the parameters of interest, rather than identifying all possible work strategies. For example, for an analysis focused on identifying topological sorts with different degrees of one-step dependencies, search algorithm could employ simple heuristics for identifying the solutions with the most and the least one-step dependencies.

Finally, it is relevant to note that the analysis so far has focused solely on characteristics of the work and the work environment, not of the team or the agents that comprise the team. Thus, any results from this chapter are a direct consequence of the work itself, regardless of the team composition, or how the work is distributed between team members. The network analysis and simulation can then identify where further work dynamics are created by the division of work within the team. Some of this may arise from an interaction between the allocation of taskwork and the dynamics within the taskwork, such as identifying cases where two different agents – perhaps in different locations – need to use the same tool. Other dynamics will further arise due to the need for communication and coordination.

4.7 Summary

This chapter introduced a formative and computational work modeling approach that can support designers of teams for complex work domains by identifying and designing for multiple work strategies. The network analysis can provide an informative assessment of a team's flexibility in choosing different work strategies. Using graph representations of work models, a range of feasible strategies can be identified as "feasible action sequences" based on linkages between states (resources) and actions. This model of work can additionally be extended to computational simulation that can estimate temporal dynamics of the work, both confirming the feasibility of the strategies identified by the network analysis and providing a detailed prediction of the time series of activity that will be required to execute the activity.

The network analysis and simulation described in this chapter comprised a first step in examining primarily the taskwork that the team must collectively perform. Once the specific work strategies (or attributes of a wide range of feasible strategies) are thus identified within the taskwork, subsequent design decisions can then examine the team composition, and the allocation of work within the team, by which this taskwork will be conducted. The strategies analysis presented here does not yet account for the synchronization opportunities and the work dynamics associated with teamwork when work is managed in multi-agent teams. Thus, the next chapter will explore methods to evaluate the costs and benefits of each strategy in a multi-agent team, accounting for dynamics and emergent behavior of teamwork.

CHAPTER 5. EMERGENT TEAMWORK IN WORK STRATEGY ANALYSIS

So far, the analysis in previous chapters considered only inherent characteristics of the work and the work domain. The previous chapter introduced an approach for identifying work strategies through the analysis of the inherent structure of a team's collective taskwork. This analysis identified work that can occur in parallel per the dependencies between taskwork and resources. However, it did not yet account for the opportunities and constraints of working in a multi-agent team, including the required inter-agent coordination of the work. Thus, following the identification of feasible strategies, this chapter elaborates on work strategies and dynamics in a multi-agent team.

As identified in the literature survey, in the conceptual design of teams there is a need for more systematic methods that can help designers assess the effect of their design decisions, particularly accounting for temporal aspects of work in a team. This chapter will focus on determining the allocation of authority and responsibility of work in a multi-agent system. Authority is the notion of which agent is executing the work, whereas responsibility is the notion of which agent is legally or morally accountable for the outcome of an action. The concepts introduced in the previous two chapters are extended here to analyze this problem in a systematic manner.

5.1 Emergent Teamwork Requirements

In a multi-agent team, work can be executed by different agents in parallel, but must abide by the dependencies discussed in the previous chapters. To manage dependencies in the

team's collective work, team members need to coordinate with each other. As discussed in the literature survey, coordination can be defined as the interweaving of activities in a team (Gary Klein et al., 2004). These coordination requirements are determined by the work allocation – what and how one needs to coordinate depends on the subset of the work that one is allocated.

As dependent actions are allocated to different agents, their dependencies create interdependencies between agents. The dependencies can create several types of coordination requirements. First, when actions are dependent through information resources, agents need to share information with each other. Thus, depending on the direction of the dependencies, there is an implied requirement to exchange information between two or more agents. Some of this information exchange might not require any overt actions. Other exchanges, however, could require action on part of the information creator (e.g., explicitly posting information for others to see) and/or on part of the receiver (e.g., translate from a displayed format to the representation needed for an action). Some exchanges could require synchronous communication between agents. The type of information exchange has consequences for the temporal dynamics of the team's collective work.

Second, when actions are dependent through physical resources, agents need to share those physical resources with each other. Where different agents use the same physical resource, there is an implicit need to physically transfer the resource between them. If the physical resource is immovable, the agents need to wait for each other as only one agent can use the physical resources at a time. With a movable resource, transfers could require the agent that last used it to bring the resource to the agent next using it or return it to a

common location. Likewise, a transfer could require an agent to fetch the resource, or, similar to information resources, it could require synchronous action for the two agents, in which they hand over the resource immediately.

Thus, the required coordination of dependent work creates additional work to be performed, here referred to as teamwork. This teamwork can result in additional activity required of the team members (Feigh and Pritchett, 2014). Thus, in the allocation of work in a multi-agent team, it is important to consider the coordination requirements associated with the allocation of authority and responsibility and the overhead in terms of the teamwork needed to fulfil these requirements. At the onset, in the modeling approach laid out in this thesis, such teamwork can be accounted for implicitly by considering the required sharing of information and physical resources, as driven by the dependencies between the taskwork and resources. Accordingly, section 5.2 elaborates on how the graph representation of the taskwork can be used as a basis for work allocation in multi-agent teams. A demonstration is provided in section 5.3.

However, teamwork not only creates overhead in terms of taskload for the team members, but also manifests in additional dependencies between the work, the work environment and the agents, which in turn have an effect on the work dynamics. Thus, it is argued here that, in good team design, there is a deliberate consideration of dynamics of teamwork and how this affects the team's performance. This includes an explicit consideration of how different forms of teamwork can be employed by a team to satisfy coordination requirements, each having its own dependencies with the rest of the work. Section 5.4 discusses how different forms of teamwork and their dynamics can be explicitly accounted for in this modeling approach, with a demonstration in section 5.5.

5.2 Analysis of Networks to Inform Work Allocation

Historically, methods for work allocation in multi-agent teams has considered agent capabilities as a basis for design. Of course, candidate work allocations should consider any limitations on which actions can only be performed by specific agents. Taking a broader view than limited categorizations such as the "Men-are-better-at/Machines-are-better-at" (MABA-MABA list) (Fitts et al., 1951), these limitations may arise from agent capabilities (e.g., only specifically trained teammates may perform specific inspection actions). Further, they may also arise from other aspects of the team design (e.g., only the commander of a space vehicle may be authorized to make mission-critical decisions). Similarly, agent location or the information available to agents may constrain which actions they can perform.

Beyond these basic constraints, however, it is proposed to identify candidate work allocations by also examining the work dynamics as captured by inherent clustering of actions in terms of information resources, physical resources, location, and dynamic precedence relationships. Where information resources are shared between agents, the network analysis identifies the information requirements of each agent and the information they will need to communicate. Likewise, with temporal and sequential dependencies between actions, the network analysis identifies which actions will require coordination. Further, when physical resources need to be shared the network analysis identifies the constraints the sharing and transfer will place on the team. Finally, several types of information can be overlaid onto the network visualizations. For example, the location of actions and resources can be overlaid to show how the team will need to position themselves and their resources through time. Knowledge of such clusters can help

designers in determining effective candidate work allocations, as these clusters reflect inherent characteristics of the work.

For a candidate work allocation, coherency defines the degree to which work clusters are allocated to individual agents. In a high-coherency work allocation, the clusters are kept intact, reducing the number of interdependencies between the agents and allowing each team members to act fairly independently within the boundaries of these clusters. A low-coherency work allocation instead breaks up the clusters across multiple agents, thereby creating interdependencies between the agents in terms of the cluster components, potentially requiring additional teamwork to coordinate and manage the sharing of the cluster components; depending on the domain this allocation may be purposely chosen to create error-checking between agents.

Several types of clusters and associated coherency can be identified to inform work allocation. First, in the information resources graph, clusters of actions and information resources reflect regularity in the work in terms of actions that work on the same aspects of the work environment. When allocating work, one can strive to maintain this regularity by allocating entire clusters in information resources to the same agent, thereby minimizing the required information exchange. In an alternative strategy, however, one could purposely break up these clusters to maximize required information exchange as a means of fostering cross-checking (Bhattacharyya and Pritchett, 2017). Likewise, this same graph can be used to identify clusters that form sequences of actions, based on directional dependencies. One could strive for coherency in these sequences when allocating the work, creating work flows wherein agents can work in parallel and relatively independently.

Second, the physical resources graph can be used to identify clusters of actions that use the same physical resources. Allocating these clusters of work to single agents reduces the required sharing and associated hand overs of physical resources. Such a work allocation strategy would be desirable when transfers are particularly costly (e.g., due to excessive weight or size of physical resources). Similarly, actions can be grouped according to the location in which they will be performed, an analysis useful when the translation between locations is costly in some way and thus should be minimized.

Third, clusters can be identified in terms of actions that are associated with the same values and priorities in the ADS, or that work towards the same goal. This is clustering up the abstraction hierarchy. Allocating such clusters to individual agents could be desirable for creating clearly defined "roles", wherein each agent has a clearly demarked contribution to the purpose of the work system. For example, on the flight deck the "Pilot Flying" is in charge of controlling the aircraft, and "Pilot Monitoring" or "Pilot Not Flying" is in charge of communication with air traffic control or other aircraft. Breaking up such clusters, however, could be beneficial when striving for creating systems with shared roles, wherein team members share responsibility for all facets of the work system.

Coherency of one type does not necessarily coincide with coherency of another type. In fact, the different types of coherency could be competing, requiring trade-offs by the designer. For example, in air traffic management, coherency in information does not coincide with coherency in location: work to control individual aircraft is (with current state-of-the-art) constrained to take place on the flight deck, but information about surrounding aircraft is most available to the controller. Hence, there is a need for information exchange between controller and pilots.

As a design objective, a coherent work allocation would allow team members to work independently, each agent having their own information and physical resources to work with/on, and own clearly delineated goals or role to focus on. However, coherency is not always possible, nor is it desirable in all cases. Other factors can limit the attainable coherency, such as work allocations limited by the inability of particular agents to work on certain locations or perform certain actions. In addition, in particular work situations, closely collaborating teams in which work and agents are highly dependent on each other can be more desirable. For example, during take-off and landing, the work of both pilots is interdependent for safety and situational awareness purposes.

Likewise, this analysis can be used to study how changes in one agent's actions (e.g., adaptation of work strategies, or failure to complete an action) can have ripple effects for the rest of the team. In a team with a coherent work allocation, each agent would be able to operate more or less independently from other agent's and as a result would be able to adapt locally without much impact on the rest of the team.

5.3 Case Study I: On-Orbit Maintenance

This case study demonstrates how the graphs of physical and information resources can be used to identify clusters in the work model, and how different types of coherency in a team's work allocation create different dependencies and resulting work dynamics. The focus will be on the team's taskwork; the second case study of this chapter will demonstrate different forms of teamwork and their effects on work dynamics. This case study is a continuation of the scenario on inspection and maintenance of panels, as discussed in chapters 3 and 4, but here the scenario is extended to include inspection of wiring and filter

components on the exterior of the spacecraft. The extended ADS of the taskwork is shown in Figure 5.1.

As in the simpler scenario, the upper-level mission objectives are split into three goals and priorities, followed by six generalized functions at the middle level: dock interaction, locomotion, inspection, panel replacement, filter replacement and wiring repair. Within these functions, the work is broken down into physical functions and, at the lowest levels, the physical and information resources. The physical functions here are defined as elemental actions, reflecting the activities each occurring at a distinct time by single agents.

Within this ADS, several clusters of work can be identified. The most obvious clusters follow directly from how the generalized functions are broken down into actions: the actions can be grouped by which generalized function they achieve. This is a grouping down the abstraction decomposition.

Other types of clustering are created by going up in the abstraction decomposition. Here, actions that are linked to the same physical or information resources can be grouped together. Figure 5.2 shows the actions (color and no outline, numbered) and physical resources (gray and black outline, lettered) as nodes, and the edges represent linkages in that these resources are required to complete the actions. The numbers and letters correspond to actions and resources in Figure 5.1. The highlighted groups are actions belonging to the same generalized function. Thus, based on there not being any overlapping parts of the shading, one can see that there are no shared resources between the generalized functions.

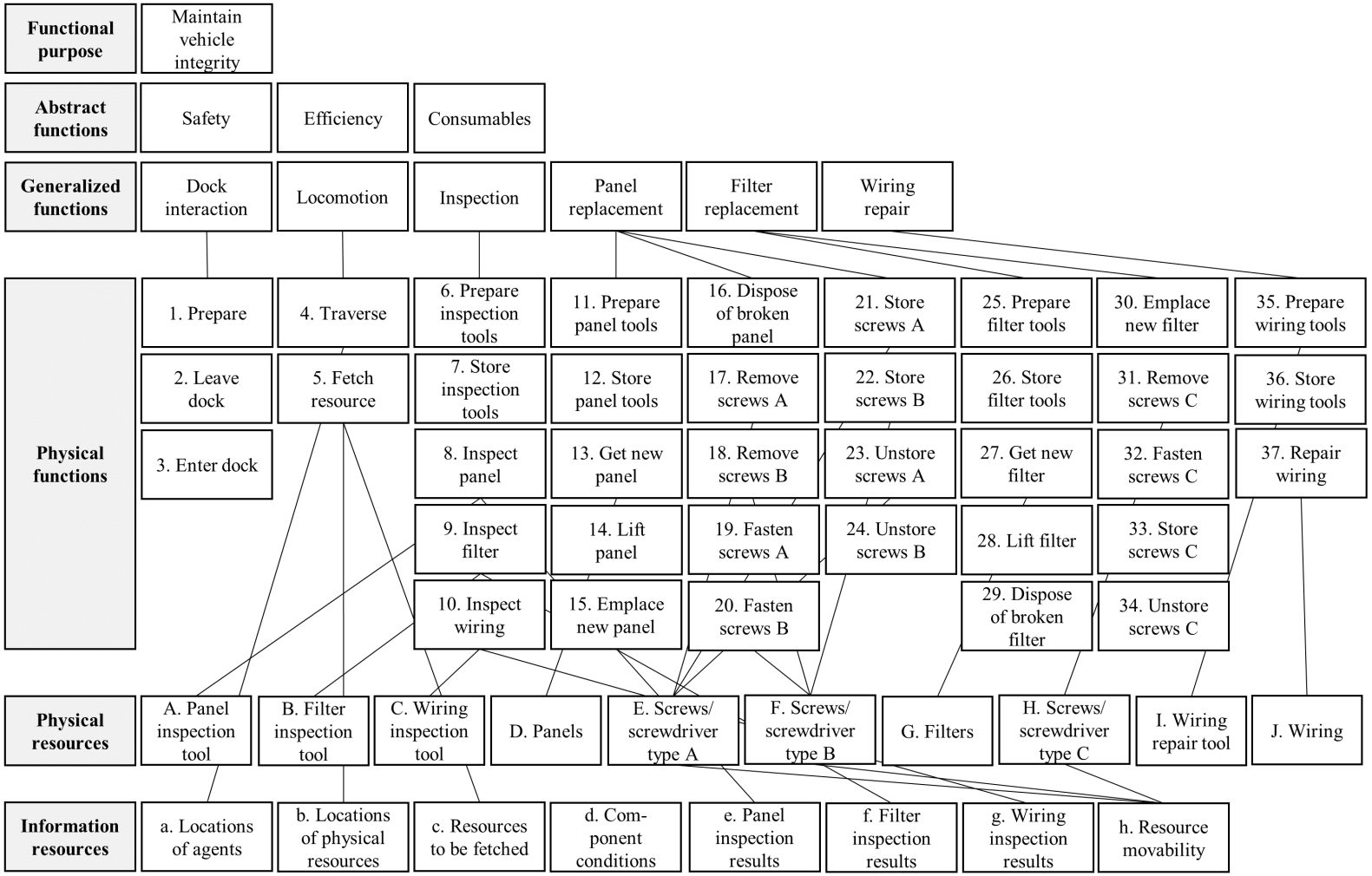


Figure 5.1: Extended ADS for the on-orbit maintenance scenario.

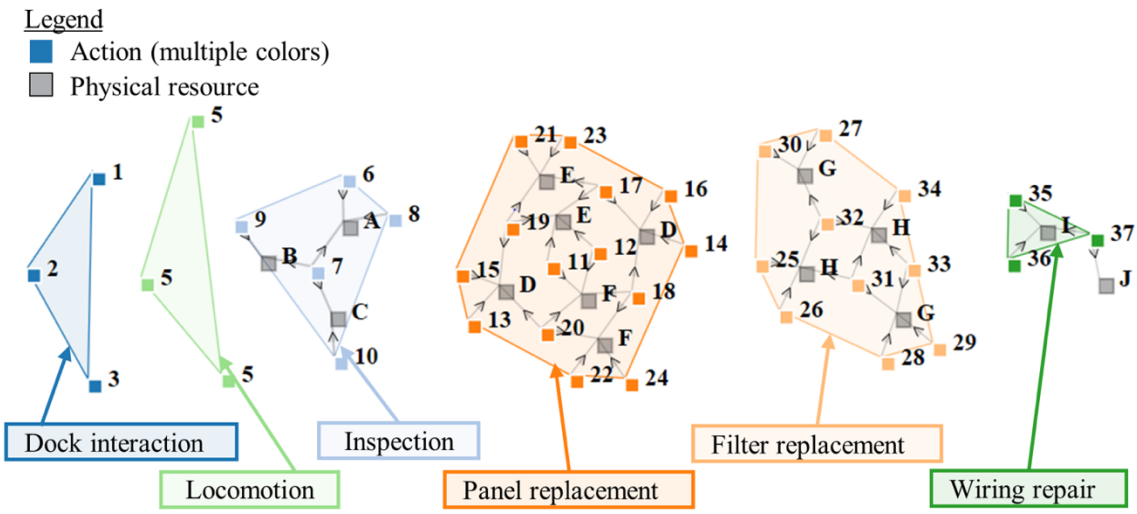


Figure 5.2: Clusters of actions and physical resources.

However, broader clusters associated with information resources do go across functional boundaries, as shown in Figure 5.3. Again, the nodes now represent actions (color and no outline, numbered) and information resources (gray and black outline, letters). The highlighted groups again are the actions belonging to the same generalized functions. It is clear that many functions share the same information resources. For example, the inspection of a panel and panel replacement share information resources associated with which panel needs to be replaced. Other functions are standalone, such as agent locomotion to fetch physical resources.

Other clusters of interest include actions that are performed at the same location, or actions that are linked through precedence relationships. In this scenario, clusters based on location are dependent on how the work evolves: the result of an inspection determines what actions will need to be performed at a specific location. For example, as inspection

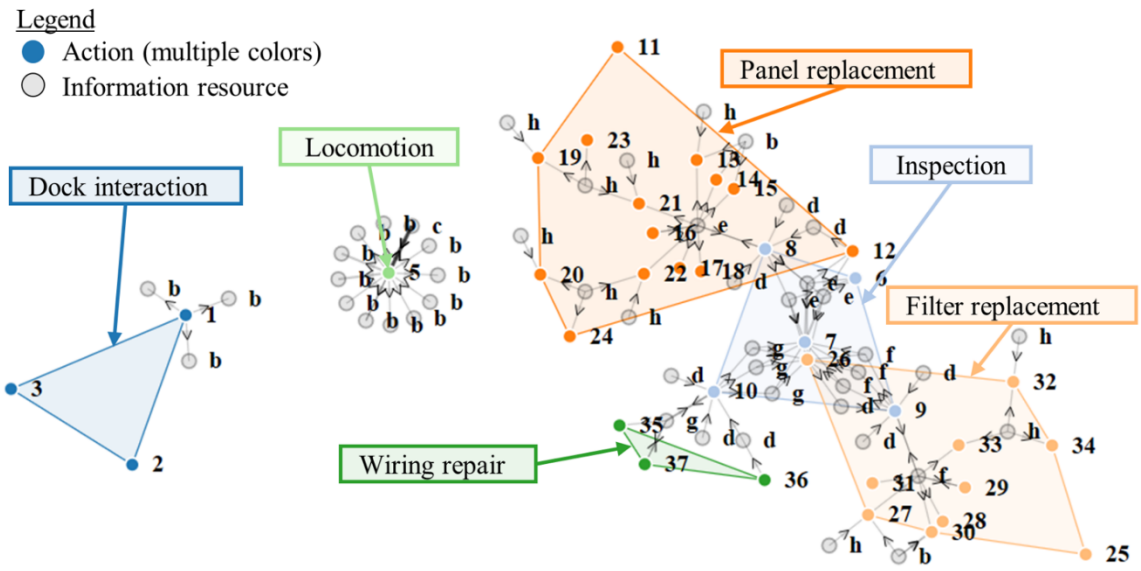


Figure 5.3: Clusters of actions and information resources.

of the panel returns that a panel needs to be replaced, the inspection action and the panel replacement actions form a cluster. In terms of clusters based on precedence relationships, the inspection actions form a cluster of preparing tools, applying them and storing the tools. Likewise, replacement or repair of components (panels, filters and wiring) each involve a sequence of preparing tools, applying them in some manner and storing them again, forming a cluster.

Based on an analysis of these different clusters within the scenario's work model, three work allocations were identified that each value a different type of coherency, see Figure 5.4-Figure 5.6. Color-coding and hashing indicate which agent has been allocated the authority for an action. Any blank blocks are shared between the agents, such as traverse and fetching actions, which are performed by all agents. It is emphasized that in a full application of the methodology many more work allocations can be defined, based on a

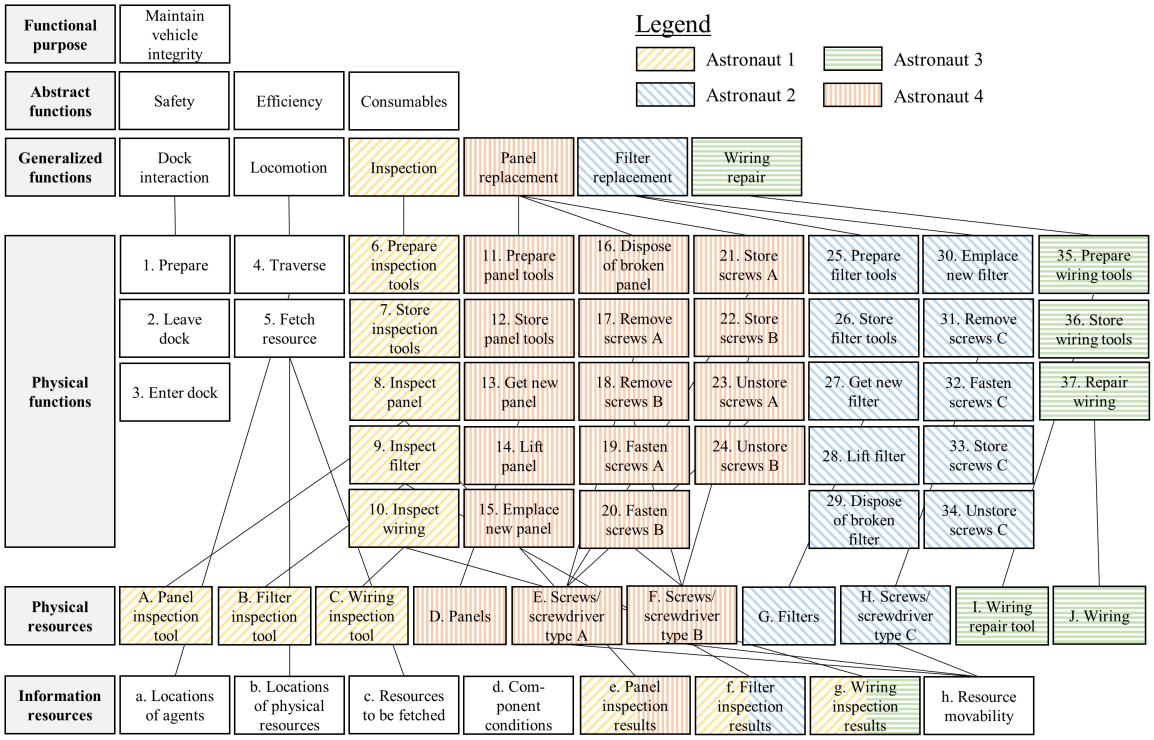


Figure 5.4: Work allocation 1, with coherency based on generalized functions.

broader analysis of the clusters. However, this demonstration is purposely limited to three work allocations specifically chosen to highlight different forms of coherency in the work clusters.

The first allocation, in Figure 5.4, strives for coherency in terms of abstracting the work according to the sequences of work providing generalized functions, such that each high-level function is allocated to a single agent. Thus, the Astronaut 1 does inspection, Astronaut 2 does filter replacement, Astronaut 3 fixes the wiring, and Astronaut 4 does panel replacement.

The second allocation, in Figure 5.5, considers cluster of physical resources and information resources. Whereas in the first allocation some information resources are

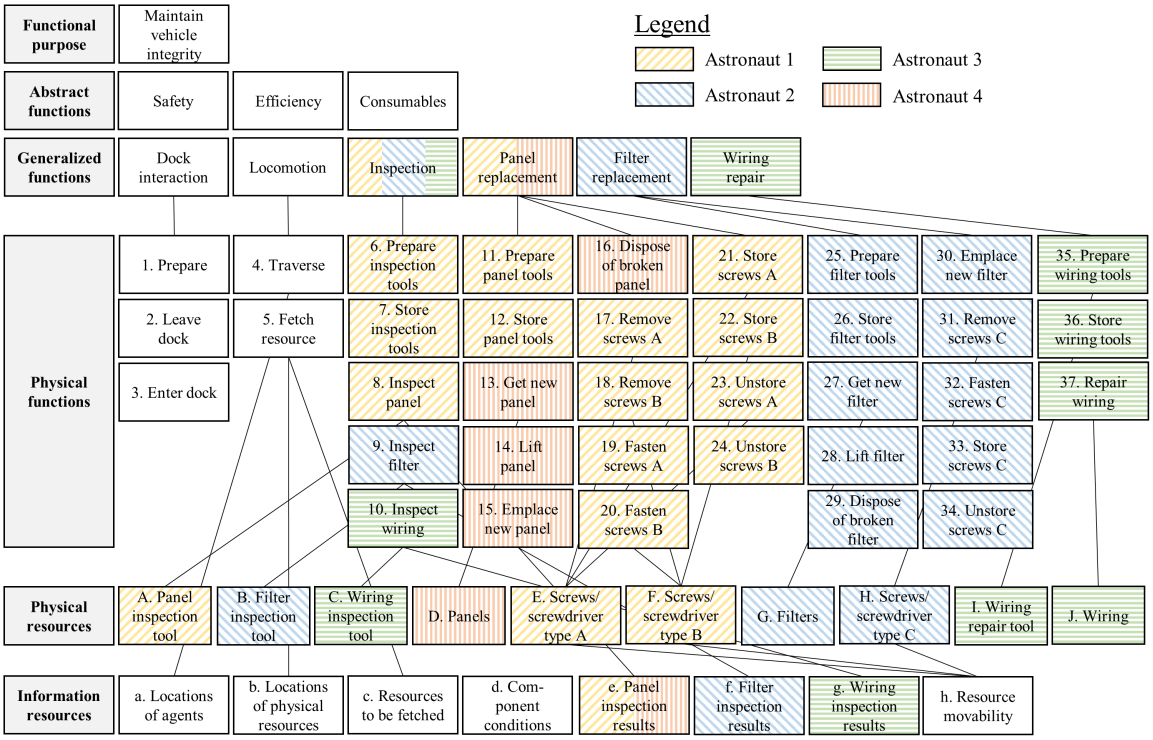


Figure 5.5: Work allocation 2, with coherency in physical and information resources, and some opportunities for parallel work.

shared between agents, this second allocation makes sure that actions that use the same physical or information resource are allocated to one agent only. For example, all actions that use the same toolset (preparing, applying to remove and fasten screws and storing the toolset) are allocated to Astronaut 1, reducing the extent that agents' actions must be coordinated around the handoff of physical resources. For coherency in information resources, the agent that inspects a component is also involved in the replacement or repair of that component, minimizing the extent that team performance is dependent on communication. For example, Astronaut 3 inspects the wiring and also does the complete repair of that component, rather than needing to communicate the result of the inspection in a manner sufficient to ensure another agent repairs it appropriately. The panel repair is

the only exception: because the panel is sufficiently large that it needs to be moved by two astronauts, this activity is distributed over these two agents and the information resource "panel inspection results" is shared between them. Of note here, by striving for coherency at the lowest level of abstraction, higher levels have become incoherent by breaking them up and distributing their work over multiple agents.

The third allocation, in Figure 5.6, explicitly strives for parallel activity whenever the work allows, to minimize mission duration which itself implies reduced risk due to shortening the astronauts' time outside the vehicle, minimized consumables, and quicker resolution of any failed components. Thus, generalized functions are broken up and parts allocated to two or three different agents. There is still coherency in terms of physical resources (when the goal is parallel work, it would be inefficient to frequently translate and transfer physical resources between agents). For example, actions that all use panels (get new panel, lift panel, set down panel, dispose of broken panel) have all been allocated to a single agent, but other repair actions have been allocated to other agents such that processes can occur in parallel. In terms of both information resources and generalized functions, this allocation is very incoherent.

The work model shown in the ADS earlier in Figure 5.1 was implemented as a collection of action and resource models within the WMC framework. In this case, four agents in the team are also represented by agent models in WMC. The three work allocations are each input to different simulation runs, defining how the actions and agents need to be linked through authority and responsibility.

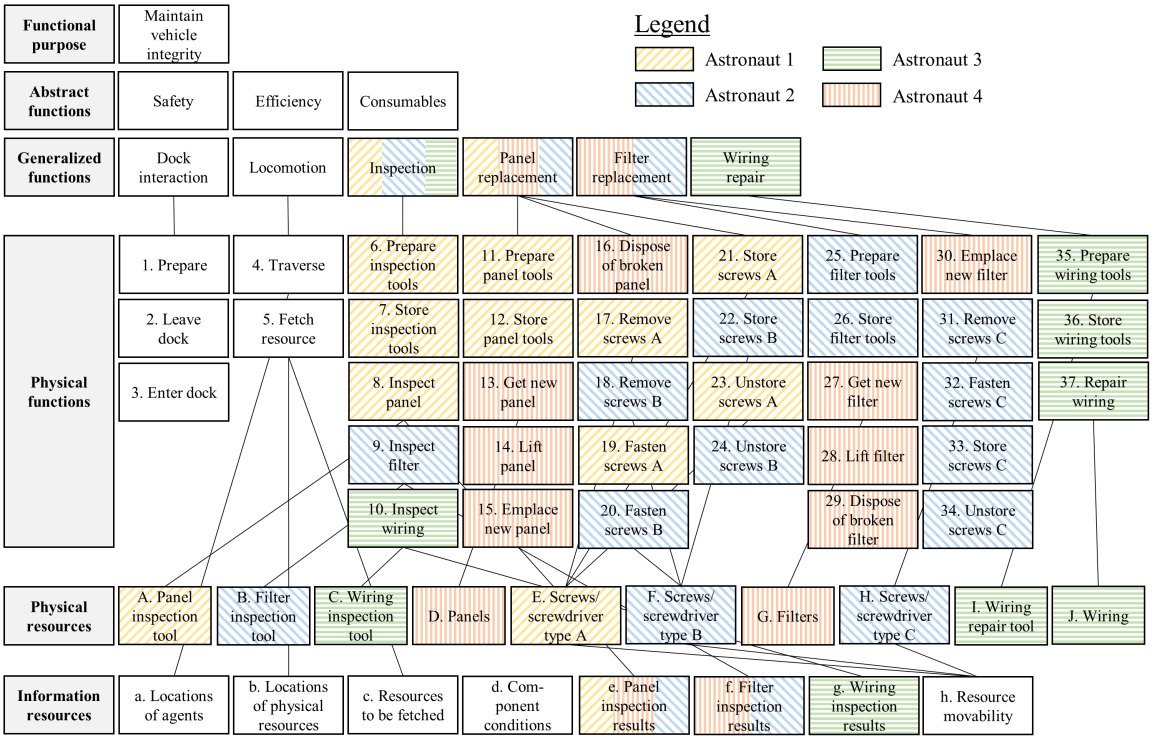


Figure 5.6: Work allocation 3, created to accommodate parallel work to reduce mission duration.

The timelines of each agent's activity in this scenario with work allocation 3 are shown in Figure 5.7. The timelines of the other work allocation are included in Appendix A. The grayscale actions are taskwork actions, as originally identified in the ADS. The color actions are traverse and fetch action, automatically identified and scheduled by the simulation to satisfy physical resource constraints and locational constraints. These results reveal that there are many instances in which agents need to wait on each other to complete their work. There are a couple of interesting work patterns that are worth highlighting. There are several instances in which physical resources need to be fetched by the astronauts, and agents need to wait for these resources to arrive before they can continue

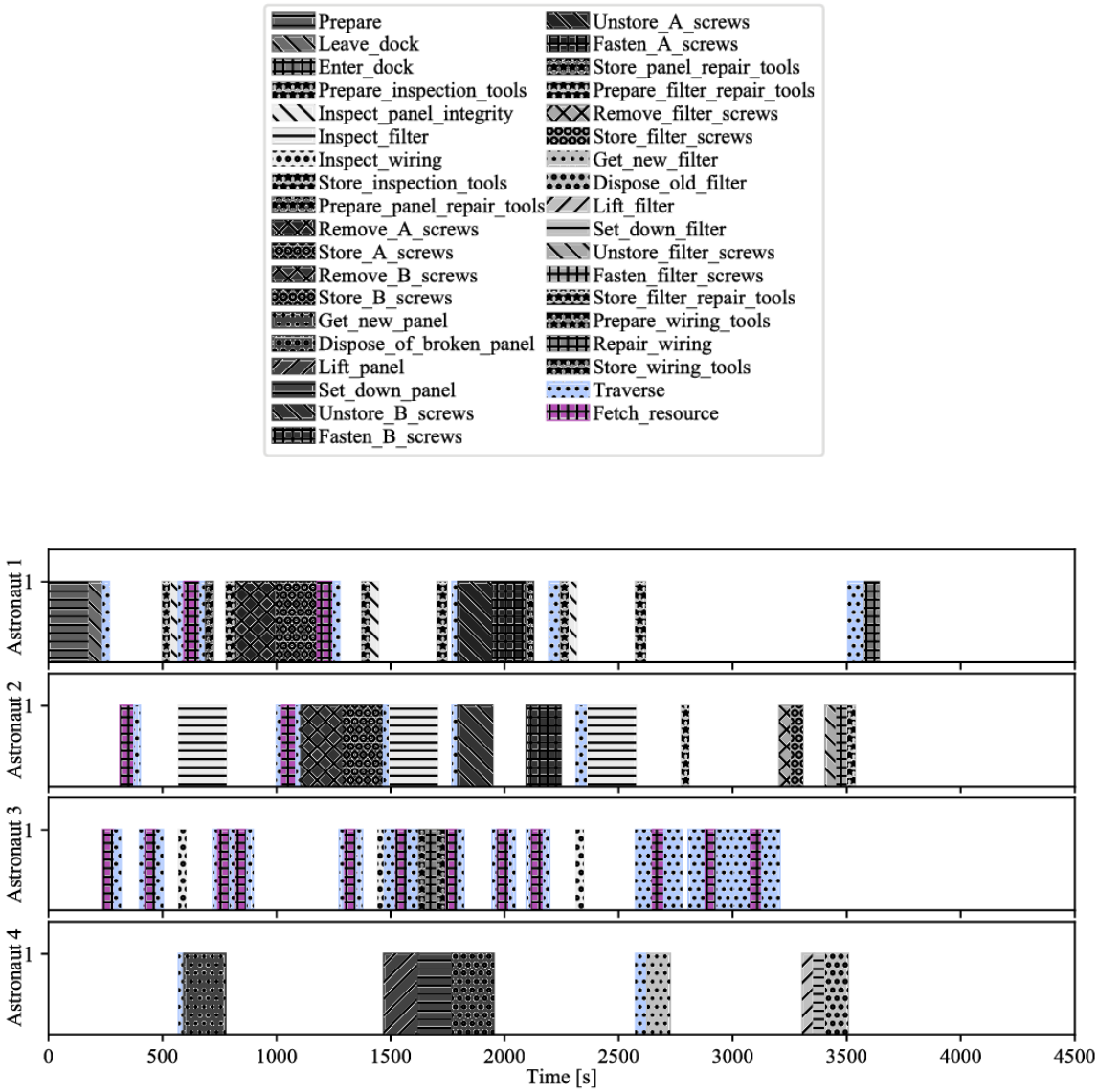


Figure 5.7: Timeline of work dynamics for work allocation 3.

their work. For example, after around 2600 seconds, astronaut 2 is waiting for astronaut 3 to complete fetching operations.

Figure 5.8 compares the three work allocations in terms of busy time (Figure 5.8a) and idle time (Figure 5.8b), with each agent's contribution shown separately. Mission durations were 57 minutes for work allocation 1, 73 minutes for work allocation 2, and 61 minutes for work allocation 3. Work allocation 1 has the shortest mission duration and idle time, indicating that maintaining functional clusters has led to an efficient interweaving of the team's work. Work allocation 2, striving for coherency in information and physical resources has the longest mission duration of one hours and 13 minutes. Considering idle versus busy time, the long mission duration seems to be attributable to the agents idling for relatively long periods. This would be expected, for this work allocation did not strive for creating opportunities for parallel work. Surprisingly, even though work allocation 3 was created to enable parallel work, the interdependencies required to coordinate the agents' parallel activities add to its dynamics such that it does not have the shortest mission duration. It does, however, show each agent contributing equally to the total idle and busy time, in contrast to the other work allocations, indicating that the work is distributed comparatively evenly across agents.

Figure 5.9 shows the total taskload, split into taskwork and different forms of teamwork actions. All work allocations look fairly similar, with work allocation 1 showing a slight reduction in required traversal. Work allocation 3 shows the highest number of required traversal actions. This high traversal load can be attributed to incoherency in geographical location, with Astronaut 1 needing to go back and forth between different inspection and

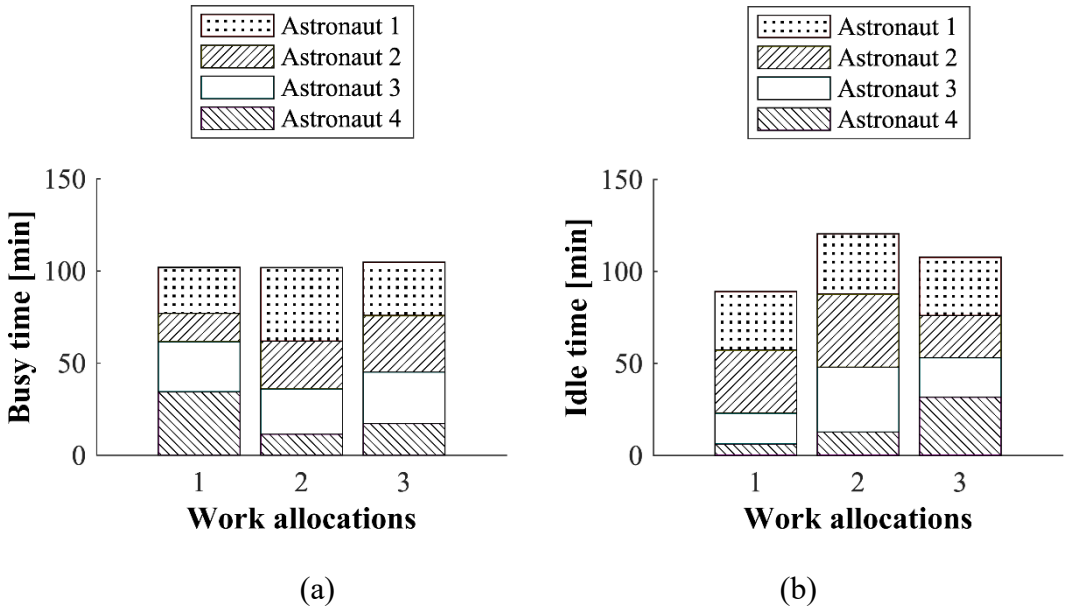


Figure 5.8: Comparison of the work allocations in busy and idle time.

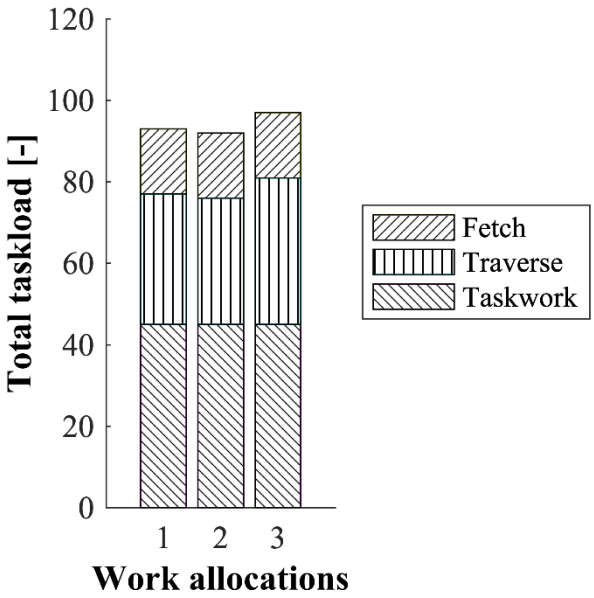


Figure 5.9: Comparison of the work allocations in total taskload.

repair sites, as well to incoherency in physical resources, which often need to be shared between agents and therefore fetched from different locations.

Figure 5.10a shows the required exchange of information resources between agents as a measure for the required communication. Information requirements at the same time and between the same two agents are represented as one information exchange in this figure. Figure 5.10b shows the required transfer of physical resources, as a measure for the required physical interaction between agents. Work allocation 1 shows the highest number of required information exchanges, which is due to its incoherency in information resources. Work allocation 3 shows the highest number of physical resource handovers. Work allocation 2 shows a low number of both information exchange and physical resources (although not lowest for both), which is to be expected given that it was specifically created with coherency in resources in mind.

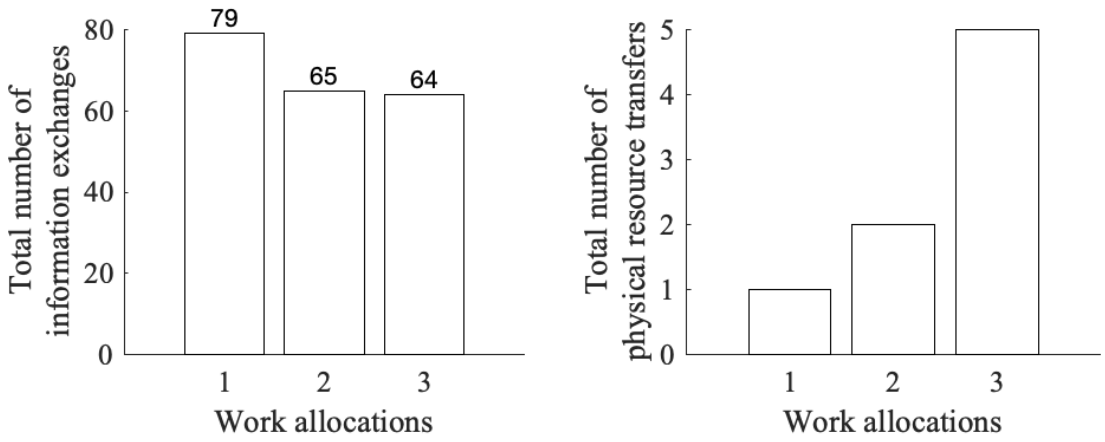
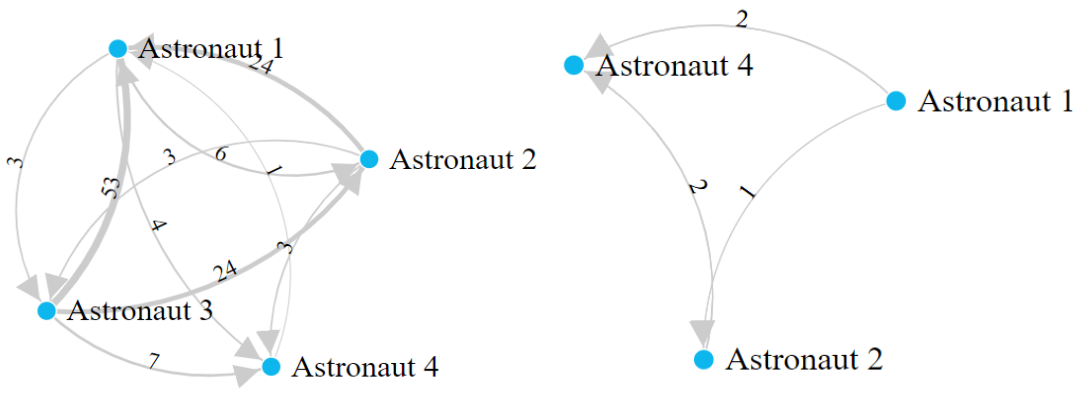


Figure 5.10: Required exchange of information and transfer of resources.



(a) (b)

Figure 5.11: Information exchange requirements (a) and physical resource hand over requirements (b) for work allocation 3.

Finally, Figure 5.11 demonstrates a visualization of the information exchange requirements (Figure 5.11a) and the physical resource hand over requirements (Figure 5.11b) between the agents as a network graph for work allocation 3. The nodes represent the agents, and the edges represent the number of requirements between the agents (the thicker the line, the more requirements). Here, one can identify sub-teams within the larger team: for example, astronauts 1, 2 and 3 seem to be relatively interconnected, whereas astronaut 4 seems to be more independent. Astronaut 3 has connections with all other agents, but information is "pushed" mostly one-way (towards the other astronauts). Likewise, Astronaut 1 is mostly receiving information from other agents, almost never needing to send any information.

5.4 Dynamics of Teamwork

As discussed briefly in section 5.1, as work is allocated to multiple agents in a team, inherent characteristics of the work and the agents creates a need for additional teamwork

to coordinate dependent work. However, because this teamwork can create considerable overhead to the team members, and also add dependencies that have an effect on the work dynamics, a good design process explicitly accounts for this teamwork in analyzing work strategies and work allocation. Often, however, the effects of this teamwork on the team's performance is emergent for different work allocations, as it is difficult to predict how the dependencies between taskwork and teamwork affect the temporal behavior of the team. Thus, this section discusses different types of teamwork and how they can be accounted for in the work model and simulation of work dynamics.

Different forms of teamwork are determined, first, by the inherent characteristics of the work, the work domain, and the agents. These characteristics include the dependencies in the taskwork that translate into interdependencies between agents, how these manifest themselves over time as the team interacts with the work domain, and the characteristics of agents in terms of their capabilities and limitations. Second, forms of teamwork are determined by the interaction modes supported within the team. If multiple modes are supported, the team has flexibility in what teamwork is performed, resulting in different work strategies for coordinating the work in a multi-agent team. However, these interaction modes can be limited by the mechanisms for communication between the agents. For example, team members can be distributed spatially, requiring teamwork to be conducted over radio or datalinks. In other, more extreme cases, the agents can be separated temporally, with communication delays preventing any synchronous teamwork. Thus, the supported interaction modes can limit the feasible forms of teamwork, and the available strategies for coordinating the team's taskwork.

For an explicit consideration of different forms of teamwork and their relations to supported interaction modes, several broad categories of coordination requirements and associated teamwork can be identified. First, there are requirements to observe other team member's actions to maintain awareness of the actions of other team members, of when dependent actions are started or completed, and of what the outcomes of these actions are.

Observations of other agent's actions is particularly important when work of agents is highly interdependent, requiring them to fluently synchronize their actions. In addition, these requirements follow from authority-responsibility mismatches, in which the agent responsible for the outcome of the taskwork is not the agent executing the taskwork; in such cases, there is an inherent requirement for the responsible agent to check the outcome of the authorized agent's action. Likewise, observability requirements might arise from safety concerns, mandating agents to cross-check each other's actions. For example, in Crew Resource Management on the flight deck, cross-checking is formalized in procedures that require pilots to communicate and confirm safety-critical information with their colleagues to create buffers for catching errors, through a set of deliberately designed teamwork protocols.

The teamwork associated with these observability requirements will be referred to as verification. Verification implies that the observing agent gets at least one information resource that is set by the observed agent. Thus, in terms of the ADS, there is at least one additional dependency between the taskwork action of the agent that is observed, and the teamwork action needed for verification. This dependency can require synchronous action between the two agents, wherein one agent verifies or observes the other agent in real-time, or it can be a dependency that is fulfilled in an asynchronous manner, wherein the

observation is performed after the observed agent completes the taskwork action. Figure 5.12 illustrates the teamwork for verification for these two interaction modes.

Second, there are requirements to direct other team members. Examples of these requirements include a need to direct other team members to perform the actions necessary to abide by dependencies, following the completion of actions that are pre- or post-dependent through information or physical resource dependencies. For example, if physical resources are shared between agents, and an agent needs to request another agent to bring over such resource, there is an implied directability requirements. Or, in hierarchical teams, a centralization of decision authority and/or responsibility can warrant a commander or supervisor agent to direct other agents, with corresponding directability requirements.

The teamwork associated with directability requirements will be referred to as control. Control implies at least one dependency between the taskwork action and the teamwork action in which the teamwork sets an information resource that is then gotten by the taskwork action. Again, these dependencies can be fulfilled in real-time, as a synchronous activity in which the directing agent provides continuous input to a team member, or as an asynchronous activity, in which the agent executed the taskwork is directed prior to commencing the action. Figure 5.13 illustrates control for these two interaction modes.

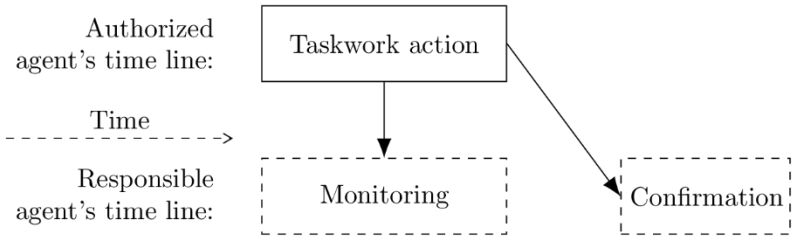


Figure 5.12: Two teamwork actions and their dependencies for verification.

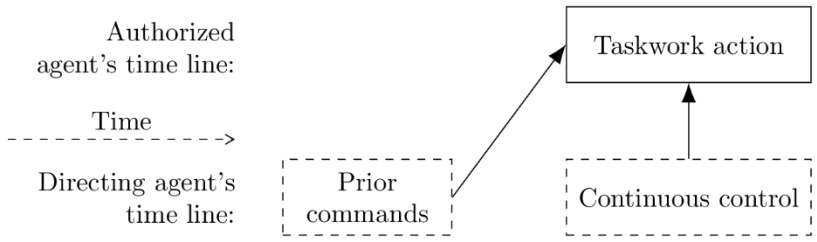


Figure 5.13: Two teamwork actions and their dependencies for control.

Teamwork actions can also have additional dependencies with other taskwork, beyond the taskwork that they are directly associated with. Teamwork for verification can be linked to post-dependent taskwork when any observability requirements need to be satisfied before beginning follow-up actions. For example, confirmation could be required before the observed agent is allowed to continue with its next action. Teamwork for control could be linked to pre-dependent actions when certain resources are required to satisfy directability requirements. For example, other agents can only be directed once the outcome of a previous action is known.

These different forms of teamwork for verification and control to satisfy observability and directability requirements, respectively, can be accounted for in the graph analysis discussed in chapter 4. Teamwork actions requiring synchronicity with the parent taskwork action can be represented as single nodes, representing both the taskwork and teamwork action, jointly executed by two (or more) agents. Asynchronous teamwork actions can be represented as their own nodes. To account for dependencies with other taskwork, additional dependencies can then be specified based on an assessment of what information is required for control, and/or what action are dependent on information from verification.

Depending on the goal of the analysis, the teamwork actions can be engendered automatically, based on a definition of the work allocation and the feasible interaction modes for verification and control, including their dependencies with the taskwork, per the definition of each mode. In addition, feasible interaction modes can be specified on a per team, per agent, or per action basis. For very broad analysis of a work allocation trade-space, without a need to assess the details of different coordination strategies, it can be beneficial to define "default" interaction modes. For more detailed analysis of different work strategies for managing the taskwork and teamwork, however, one can specify the interaction modes and tune their dependencies on a per-action basis.

With teamwork actions added to the graphs, the model of taskwork and teamwork and their dependencies can be analyzed for its remaining flexibility and to identify different strategies for managing both the taskwork and teamwork, using the concepts discussed in chapter 4. First, different paths could involve different types of teamwork, where different interaction modes could show up as different paths through the network. Thus, this could help identify work strategies involving different combinations of monitoring, confirmation, command or control. These strategies can then be simulated to assess the temporal behavior of the team given the additional dependencies between taskwork and teamwork.

Second, topological sorts could distinguish different control modes in terms of event-horizon: sorts can be more proactive or reactive in the degree to which future information or physical resource needs of other agents are anticipated, as different timing of the taskwork necessary for team members' actions, or as different timing of the teamwork for control and verification of other agents. For example, command actions can be executed proactively, anticipating future directability requirements, or reactively, once an agent

requires directions to continue. Likewise, outcomes of a taskwork action can be verified immediately following the completion of that action, proactively creating awareness that might be necessary for future actions, or reactively, once awareness of the outcome is required for a follow-up action.

5.5 Case Study II: Rover Dynamics

This case study demonstrates the effects of teamwork on the work dynamics, including the continuous rover dynamics modeled in earlier chapters. The team consists of two astronauts that are driving the lunar rover, identified as Astronaut 1 and Astronaut 2. For each team design, different work strategies are identified in terms of the feasible interaction modes. Two different team designs in terms of the allocation of responsibility are investigated through simulation.

The two work allocations are shown in Table 5.1. For both team designs, Astronaut 1 is allocated authority for actions associated with driving the rover, including the required checks on battery levels and subsystem temperatures, and Astronaut 2 is allocated authority for actions associated with imagery. Team design 1 has Astronaut 1 acting as the commander, providing directions to Astronaut 2. The commander is allocated responsibility for all of the team's actions and as such is required to cross-check the actions executed by Astronaut 2. From this definition of the allocation of authority and responsibility within the team, it follows that Astronaut 1 needs additional teamwork to direct and verify Astronaut 2's actions. As the interaction mode we define teamwork actions to direct Astronaut 2 prior to starting his/her action and teamwork actions to confirm the outcome after Astronaut 2 has completed actions. In an alternative interaction,

Table 5.1: Two team designs with different allocations of responsibility. A = authority, R = responsibility

Taskwork	**Team design 1**		**Team design 2**	
	Astronaut 1	***Astronaut 2***	***Astronaut 1***	***Astronaut 2***
1. Check battery levels	A/R		A/R	
2. Check temperature	A/R		A/R	
3. Assess location and attitude	A/R		A	R
4. Plan rover path	A/R		A	R
5. Estimate size of objects	R	A		A/R
6. Localize obstacles	R	A	R	A
7. Select next waypoint	A/R		A	R
8. Move rover				
9. Change camera angle	R	A		A/R
10. Capture imagery	R	A		A/R

we have Astronaut 1 control and monitor astronaut 2 during the safety critical actions (italicized in Table 5.1), as synchronous processes involving both agents simultaneously. Finally, “MoveRover” is allocated to neither agents, as it is modeled as a continuous process of the work domain.

Thus, team design 2 has a flat hierarchy with the astronauts sharing responsibility in a way that maximizes the cross-checking of the team’s safety-critical actions. The safety-critical actions are deemed “AssessLocationAttitude”, “LocalizeObstacles”, “PlanRoverPath”, and “SelectNextWaypoint”. Based on this definition, there is a need for teamwork for both astronauts to verify each other’s actions. For “AssessLocationAttitude” and “LocalizeObstacles” these cross-checking requirements are fulfilled through confirmation; for “PlanRoverPath” and “SelectNextWaypoint” the requirements are accounted for through synchronous monitoring. Furthermore, per design, to maximize the chances that errors are caught in time, cross-checks must be completed before results of

the checked actions are used for any follow-up actions that are dependent on the output of the checked actions.

Figure 5.14 shows the graph of actions and information resources for team design 1 with teamwork actions for prior command and posterior confirmation of the actions of Astronaut 1 (shown in blue). The command actions are not dependent on any information, meaning the command actions can be executed at any time, as long as they are executed before the action-to-be-commanded. Because confirmation must be completed before any follow-up action is started, there are additional dependencies between the confirmation resource and the taskwork actions that get information from the action-to-be-confirmed. A similar graph can be created for the synchronous interaction mode, which includes monitoring and control actions instead of confirmation and command actions. Likewise, graphs can be created for team design 2.

Then, from each of these graphs, different work strategies can be identified. Figure 5.16 and Figure 5.16 show feasible work strategies for team design 1 and 2, respectively, differing in the teamwork that is used to fulfill the requirements for observability and directability. It can be seen that the different forms of teamwork create very different sequences of actions, with varying options for parallel work and required synchronous work. Just as described in this case study in chapter 4, some of the paths in these strategies do not need to be executed in every iteration of the work, based on the quality of information. Thus, within these full work strategies, alternative work strategies can be found using the approach described in chapter 4.

Taskwork
1.CheckBatteryLevels
2.CheckTemperature
3.AssessLocationAttitude
4.PlanRoverPath
5.EstimateSizeObject
6.LocalizeObstacles
7.SelectNextWaypoint
9.ChangeCameraAngle
10.CaptureImagery

Teamwork
11.CommandEstimateSizeObject
12.CommandLocalizeObstacles
13.CommandChangeCameraAngle
14.CommandCaptureImagery
15.ConfirmEstimateSizeObject
16.ConfirmLocalizeObstacles
17.ConfirmChangeCameraAngle
18.ConfirmCaptureImagery

Resources
A.BatteryLevels
B.ObservedBatteryLevels
C.SubsystemTemperatures
D.ObservedSubsystemTemperatures
E.CurrentLocation
F.ObservedLocation
G.HeadingDeg
Q.GoalLocation
R.PathPlanned
S.NextWaypoint
T.RealMap
U.RockLocations
V.RockSize
W.CameraAngles
X.Imagery
aa.EstimateSizeObjectCommands
bb.LocalizeObstaclesCommands
cc.ChangeCameraAngleCommands
dd.CaptureImageryCommands
ee.EstimateSizeObjectConfirmation
ff.LocalizeObstaclesConfirmation
gg.ChangeCameraAngleConfirmation
hh.CaptureImageryConfirmation

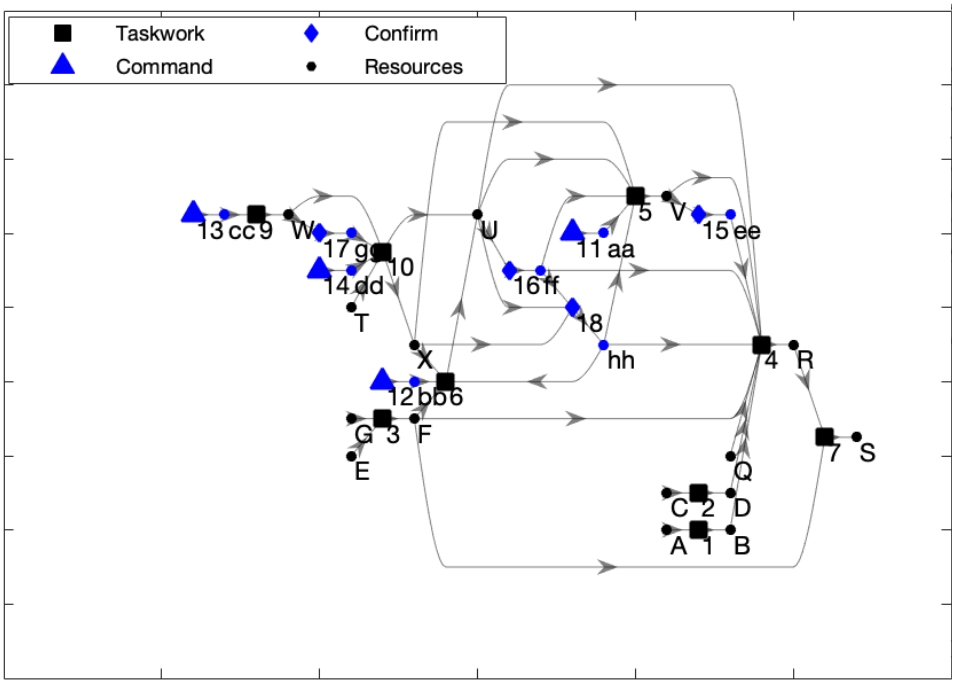


Figure 5.14: Network visualization of the taskwork and teamwork for team design 1.

Taskwork
1.CheckBatteryLevels
2.CheckTemperature
3.AssessLocationAttitude
4.PlanRoverPath
5.EstimateSizeObject
6.LocalizeObstacles
7.SelectNextWaypoint
9.ChangeCameraAngle
10.CaptureImagery

Teamwork
11.CommandEstimateSizeObject
12.CommandLocalizeObstacles
13.CommandChangeCameraAngle
14.CommandCaptureImagery
15.ConfirmEstimateSizeObject
16.ConfirmLocalizeObstacles
17.ConfirmChangeCameraAngle
18.ConfirmCaptureImagery

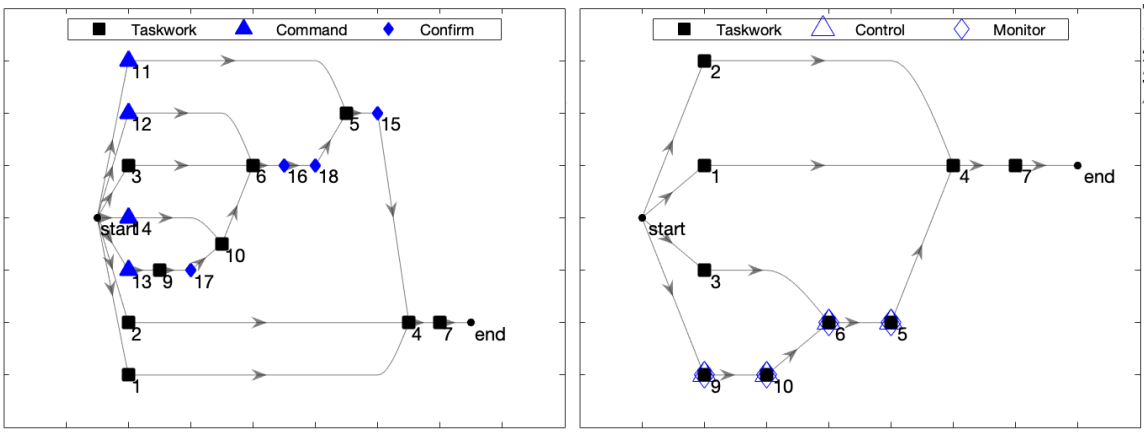


Figure 5.15: Feasible work strategies with team design 1.

Each of these work strategies can be simulated to evaluate the temporal dynamics of the team's work. Figure 5.17 shows the timelines of actions for the two work strategies of team design 2. A work strategy with confirmation to cross-check the safety-critical actions (see Figure 5.17a) results in frequent teamwork actions with intermittent idle time for Astronaut 2. Astronaut 1, on the contrary, needs to confirm the actions of Astronaut 2 between other taskwork actions, thus needing to switch attention between different work processes. When using monitoring instead of confirmation (see Figure 5.17b), idle times are reduced as both astronauts are monitoring each other's taskwork.

Taskwork
1.CheckBatteryLevels
2.CheckTemperature
3.AssessLocationAttitude
4.PlanRoverPath
5.EstimateSizeObject
6.LocalizeObstacles
7.SelectNextWaypoint
9.ChangeCameraAngle
10.CaptureImagery

Teamwork
11.ConfirmAssessLocationAttitude
12.ConfirmLocalizeObstacles
13.ConfirmPlanRoverPath
14.ConfirmSelectNextWaypoint

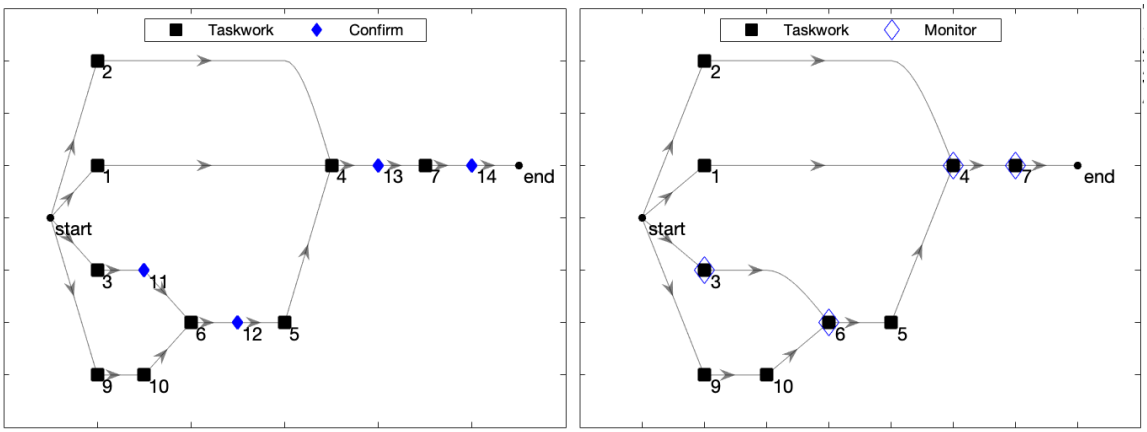
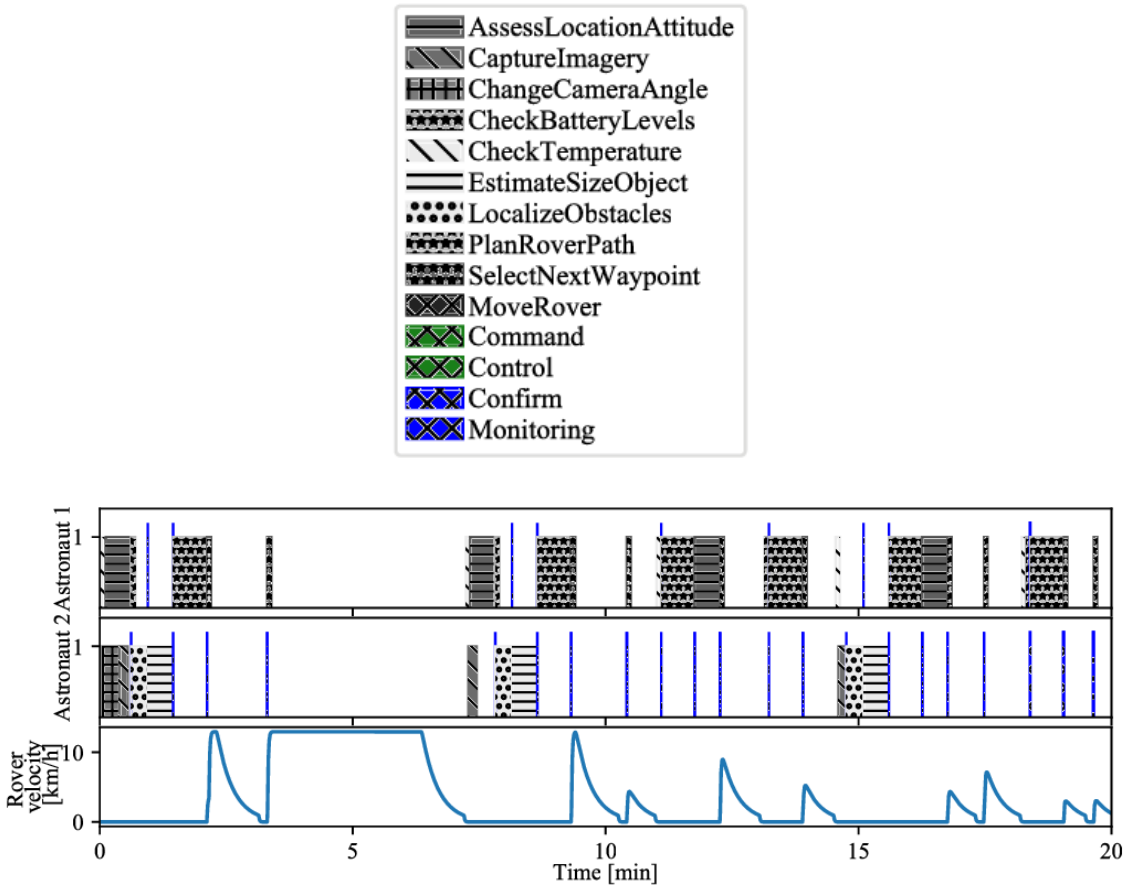


Figure 5.16: Feasible work strategies with team design 2.

The simulation results presented here were computed with a taskload limit of one for each agent. However, the teamwork actions were considered of lower demand relative to the taskwork and not accounted for in the agent's taskload. Thus, the agents were assumed to be able to monitor another agent's task even as they completed their own tasks. When an analysis shows that the monitoring is demanding to a degree that no other actions can be executed in parallel, the simulations can be run with the teamwork included in taskload considerations. In this case, different work patterns and dynamics would emerge.



(a) Work strategy with confirmation

(b) Work strategy with monitoring

Figure 5.17: Timeline of work dynamics for two work strategies with different teamwork for team design 2.

Of final note, for each of these team designs and their feasible work strategies, the work can be interweaved in different ways, representing different methods for managing the dependencies within the team. For example, in team design 1, the commander can provide directions to astronaut 2 prior to starting any other taskwork. Alternatively, directions could be provided in a reactionary manner: only when the need arises for astronaut 1 to do its taskwork does the commander provide directions. As discussed in chapter 4, these different ways of interweaving the work can be identified through alternative topological sorts. Thus, further analysis of the team's work dynamics could also identify these different orderings and simulate them to assess how dependencies between the teamwork and taskwork result in different team behavior.

5.6 Discussion

Work clusters and coherency are presented as constructs for reasoning about different design options for work allocation in a multi-agent team. The graph representation of a team's collective work can be used as a formative design tool for thinking through how work allocation affects the work's coherency in terms of higher-abstraction functions, and lower-abstraction information and physical resources, as well as work's location. This approach allows for a direct analysis of how dependencies in the work translate into interdependencies between agents and a consequential need to coordinate work within a multi-agent team. Thus, by considering the natural clusters within the work, and the different types of coherency, designers can estimate how information and physical resources are shared, how goals, values, and priorities of the team are shared, and the team's spatial management, amongst other concerns.

In addition, this approach can be extended to consider different team structures and hierarchies. For example, based on interdependencies between agents, sub-teams can be identified within the larger team, as subsets of agents that are interdependent, but have relatively little dependencies with the rest of the team. A brief example was given in case study, in which a commander provided commands to and confirmed the actions of a "subordinate" astronaut. However, one could envision more elaborate analysis of how directions and verifications might flow up and down a hierarchy through different forms and degrees of teamwork.

Different interaction modes are appropriate to different situations. From a contextual control perspective, the perceived available time affects which type of teamwork is effective. For example, when perceived time is low, a work strategy with confirmation and commands that allows quick checks between other taskwork can be advantageous over a work strategy that requires agents to monitor continuously at the expense of performing work. Additionally, interaction modes can be constrained by contextual factors such as available communication channels, such as datalinks and radio, or time delays associated with temporally distributed teams. Thus, the formative analysis of work allocation should consider multiple work strategies for managing both the taskwork and the teamwork, as can be identified within the work graphs.

Finally, in comparison with existing methods for designing teams, this work modeling approach is especially powerful as, following a static analysis, the work can be simulated to directly evaluate the temporal behavior of the team in terms of the work dynamics, including any continuous dynamical processes that the work and the team would be interacting with. In addition, the analysis is powerful in its systematicity, with the potential

to use computational power to efficiently explore the design trade-space for work allocation and interaction modes. For example, one can automatically investigate different interaction modes and their effects on the team's effectiveness, for a variety of work strategies. In principle, the design trade-space is infinitely large, so such an exploration should use heuristics that could be derived from subject-matter experts. It is up to the designer how and what part of the trade-space is explored.

5.7 Summary

This chapter discusses the emergent teamwork for coordinating dependent taskwork in a multi-agent team. With the computational models of work in teams, and the identification of feasible work strategies, discussed in the previous chapters, the cost and benefits of work strategies can be analyzed, as well as how team design affects the team's feasible work strategies. This approach is here applied to the conceptual design of teams in terms of the work allocation and the definition of interaction modes. First, the graph representations of a work model can be used to identify natural clusters in work. A work allocation can then be characterized by its coherency, defined as the degree to which work clusters are allocated to single agents. Thus, a formative analysis of work clusters and coherency can be used as a basis for work allocation.

Second, the inherent structure in a work model, together with an allocation of the work in a multi-agent team, create emergent requirements for coordination, and a consequential need for teamwork. The chapter highlights the importance of explicitly accounting for these teamwork actions and their effects on the team's work dynamics. Again, the graph

representation can be used as a formative analysis tool to investigate the effects of different interaction modes for satisfying these teamwork requirements.

CHAPTER 6. WORK STRATEGIES IN HUMAN-ROBOT TEAMS

In many complex work domain, robotic capabilities can complement the abilities of a human team. Robots have the physical capabilities to operate in harsh, unforgiving environments where human agents would otherwise be exposed to considerable risk or would be unable to go altogether. Space operations is one example; others include disaster response, nuclear energy, and search and rescue. Robots can also be used to off-load tasks from human agents, freeing them to focus on other tasks that are safety-critical, more fulfilling or otherwise more relevant to the human agents. Finally, robots can perform tasks that are considered dirty. Robots, however, often rely on the cognitive capabilities of human team members to operate the robot remotely, or to command it and then verify its performance.

From the literature survey it followed that there currently is no design method for human-robot teams that can comprehensively consider common issues with human-robot teams. Several formative methods are available, but these methods lack systematicity, and (perhaps as a consequence) there is a tendency of designers to employ simpler normative methods, such as the levels of automation paradigm. In addition, most existing methods are static, in that they do not account for dynamics of the work and the work environment.

In the preceding chapters it is argued, however, that many aspects of a team's effectiveness can only be assessed when considering the temporal component of a team's behavior. This is hypothesized to be particularly true for human-robot teams – for example,

the coordination and synchronization of activity over time is of major concern to designers of human-robot teams. Thus, further developments of methods for the design of human-robot teams can build on the conceptual ideas of earlier methods, but strive for more formative, systematic and dynamic methods that can inform designers of conceptual design decisions. Thus, this chapter discusses how the concepts and methods laid out in earlier chapters apply to the design of a human-robot team.

It is argued that the method presented in this dissertation can be used to account for common problems in human-robot teams, by allowing designers to gain insight in how their design choices might alleviate or aggravate these issues. First, it allows designers to explicitly account for the teamwork that is involved in coordinating the joint work of human and robotic agents, and, second, it allows designers to explicitly account for the temporal behavior of the taskwork and the teamwork for coordination.

6.1 Dynamics of Human-Robot Interaction

This section discusses work dynamics in relation to human-robot teams. Many of the concepts from earlier chapter can be applied here, to analyze how robotic team mates would influence the work strategies and work dynamics of a team. This chapter first extends the discussion of challenges to human-robot team design to discuss how limitations of robotic team members can impact a human-robot team's ability to adapt to changing work demands. In the literature survey, the main challenges to human-robot team design were identified as technology being rigid in its operations and technology's inability to bear responsibility for work.

First considering the rigidity of technology, a challenge to the design of human-robot teams as opposed to human-human teams is that the rigidity of technology can limit flexible coordination of taskwork and teamwork. Robotic team mates are often clumsy in their coordination in terms of the sharing of information and physical resources. They may additionally be bound to particular sequences of work, beyond inherent characteristics of the work discussed in chapter 3. The timing of actions is often rigidly encoded or implied in the robot's design, without its ability to adapt its actions and work strategies to the immediate context that the team is operating in. For example, human workers can reasonably predict when it is best to communicate necessary information. Technological agents, in contrast, have historically been clumsy in the timing of information transfer, therein lacking the ability to anticipate when information is needed by other team members.

In addition, in coordinating with other team members, humans are typically reasonably flexible and adaptive in their modes of interaction. In the case of human-robot or human-automation interaction, however, the machine is comparatively inflexible in the modes of interaction, dictating the human's activity for commanding, controlling, monitoring and/or confirming the machine. Thus, the robotic team member is not only constrained to particular action sequences, but its team mates might also be expected to make their own actions fit these rigid patterns of work, thereby limiting their ability to select work strategies appropriate to the context.

A further challenge to the design of human-robot teams is that technological team members cannot be held legally accountable for the outcome of an action. Thus, when allocating work to a robotic agent, any responsibility for the work remains with a human worker. Because humans are ultimately responsible for the robot's or automation's actions,

a human supervisor is needed to verify the outcome of these agents' actions. Thus, the human needs to intermittently or continuously monitor the robot or automation, and/or needs to confirm action outcomes before the technological agent is allowed to continue with follow-up tasks, especially when actions allocated to the robot are critical to the success of the team. This can lead to significant overhead for the human supervisor that could, first, create additional work for the human team members and, second, creates additional dependencies between the robot's and human's actions which in turn need to be managed temporally.

Related to the rigidity of technological agents, historically these agents require frequent human input for cognitively demanding or complex tasks such as judgment and decision-making. Robotic agents often do not have the ability to integrate lower-level information and account for contextual factors that human decision-makers would consider. Moreover, human team members might not accept a robotic team mate making decisions independently, especially when such decisions can have broader implications for the team's success. Finally, particularly in complex work domains, where the task demands vary and the team needs to adapt in response to context, technology has historically needed human supervisors to account for eventualities not considered in its design. Thus, the limitations of robotic team members' abilities or the unwillingness of human team members to have technology operate independently often requires joint work in which both the human and robot contribute to the execution of taskwork.

In conclusion, the limitations of robotic or automation agents (rigidity in its operation, inability to bear responsibility, requirements for joint work), at least in the current state of robotics, creates a limited team member which has implications for the dynamics of the

collective work in a human-robot team. In design of human-robot teams, the limitations and their effects need to be taken into account. The next section discusses these implications through the perspective of the modeling framework presented in the earlier chapters.

6.2 Implications for Human-Robot Team Design

If how a human-robot team adapts is determined, as hypothesized, first, by how many work strategies are available and, second, how well the attributes of feasible work strategies match the work demands of the team, the limitations of robotic team members can significantly limit a team's ability to adapt. These limitations, can, if not properly accounted for in design of human-robot teams, result in teams that are inefficient and brittle when needing to adapt to changing work demands. However, it is argued here that with an explicit analysis of the team's collective work, its dependencies and required teamwork, designers of teams can identify the effects and implications of robotic limitations, and account for potential issues with a team's ability to adapt early-on in design.

The modeling framework presented in the earlier chapters can account for the limitations of robotic team members, not just in terms of its capabilities (as would be the basis for design in traditional methods for team design, e.g., the MABA-MABA lists), but also its broader implications for human-robot teams in terms of the effects on feasible work strategies, work dynamics and the team's effectiveness. Indeed, at a high level, limited agent capabilities could render certain team designs (i.e., work allocations or interaction modes) infeasible. Additionally, agent capabilities can be accounted for in the duration of actions; compared to human agents, robotic agents could need more (or less) time to

complete an action (Akin et al., 1989). Earlier work in robot-astronaut operations allocated the tasks based on these two factors, with the goal of minimizing human involvement and idling time (Singer and Akin, 2009). As described in the previous section, however, the capabilities of robots and humans play out in more intricate ways that are often emergent and difficult to predict from a static analysis alone. Thus, here these limitations are related to the modeling concepts that are central to this thesis, with the goal of providing means to designers of human-robot team to consider the broader implications of these limited team members.

The rigidity of robotic team members can be captured in a number of ways. First, when robotic team members are able to only execute the taskwork in a limited number of ways, this constrains the dependencies between the robot's taskwork and the resources. Thus, whereas a human team member would have the flexibility to go through multiple paths within the graph, only one or a few paths may be supported within the robotic agent's design. Likewise, when the work characteristics allow flexibility in the order of the actions, the rigidity of robotic team member could constrain the feasible topological sorts to one or a few supported strategies.

Any rigidity in the robot's "parts" of the graph would also impact the rest of the team, through the dependencies between the resources shared by human and robotic team members. Thus, as the robot's path through the network is fixed, the other team members are limited in their own flexibility for choosing different paths or different orders of the actions, as they are required to interweave their activities with the rigid sequences of the robotic team member.

Moreover, robotic limitations can also have implications for the required teamwork, and to what degree team members have flexibility in choosing different forms and timing of teamwork for fulfilling observability and directability requirements. First, a robotic team member's limitations to bear responsibility for work creates additional observability requirements. Thus, there needs to be a human team members who is legally and morally accountable for the work that the robot is executing. With such authority-responsibility mismatches, as previously discussed in chapter 5, there is an inherent observability requirement on the responsible agent, to verify the outcome of the robot's actions. Second, as robotic team members are limited in their capabilities, or decision-authority is not willingly allocated to an independent robotic team member, there is a need for human and robot to perform actions jointly. This joint work creates additional observability and directability requirements. For example, with current state-of-the-art, robots need frequent directions from human team members to compensate for their limited capabilities and rigidity.

Then, to fulfill these requirements, compared to human-human teams, there is a relatively higher need for teamwork associated with verification and control of robotic team members. This creates a higher taskload for the human agents in the team, and comes with additional dependencies that limit the feasible work strategies. Moreover, robotic team members might only support a limited number of interaction modes, limiting the flexibility in different forms of teamwork. Indeed, similar to how flexibility in the team's taskwork is reduced, limitations in the interaction modes supported by robotic team mates can render certain forms of teamwork infeasible, could limit the number of paths through the graph,

and/or, with rigidity in the required order of actions, could limit the degree to which teamwork actions can be executed proactively or reactively.

Further, these relatively rigid and strict requirements for teamwork actions add a further temporal component to the collective dynamics of the team's work, which computational simulation can further predict. Thus, reflecting the relatively higher teamwork burden on the human operator in human-robot teams, the teamwork actions can be given an explicit duration, and occupy human team members as they are executing the teamwork. For example, the robot cannot perform an action until its human supervisor is free to command and monitor it – and the human may not be able to perform her/his own taskwork in parallel.

Figure 6.1 highlights different combinations of teamwork and taskwork (other combinations are possible too), showing explicit durations for the teamwork actions. For example, Fong, Zumbado, Currie, Mishkin, and Akin (2013) note that, in space operations, common modes for interacting with robotic agents include command sequencing and direct teleoperation. Command sequencing is generally in the form of detailed actuator commands, such as the number of hand rotations, which are then sent to the robot for execution. Direct teleoperation involves continuous, real-time control through a hand controller, line-of-sight and/or video links (e.g., human can directly operate a Remote Manipulator System (RMS) through a joystick, or can provide commands through a datalink and let the RMS execute preprogrammed commands).

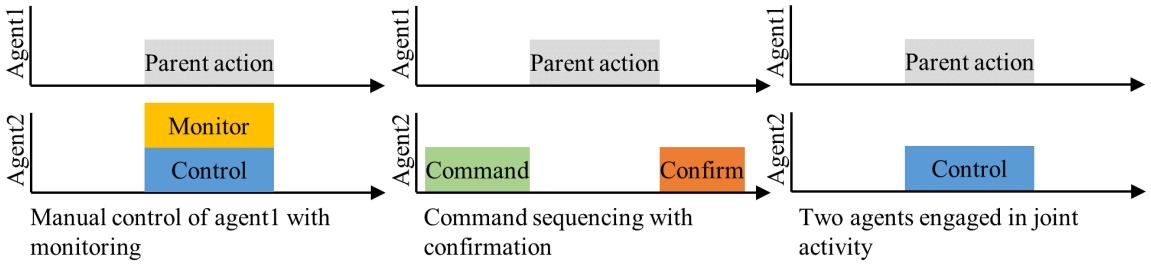


Figure 6.1: Different combinations of human-robot teamwork.

In conclusion, the prior discussion frames issues encountered in human-robot teaming in a way that allows for evaluation of their effects in a broader sense, including the temporal dynamics of a team's collective work. The next two sections continue the two case studies, considering how work strategies are affected by the introduction of robotic team members.

6.3 Case Study I: On-Orbit Maintenance

Communication delays associated with deep space will demand a shift in the operations of manned spaceflight missions. Whereas astronauts now rely heavily on real-time ground support, future astronaut crews will need to operate autonomously from Earth. To alleviate some of the burden that this shift will pose to astronauts, NASA and other space agencies envision the use of robots. While automation and robots have a long history in space operations, these technologies currently have the role of an aid or a tool for astronauts; often, the robotic systems are tele-operated from mission control on Earth. For future space operations, however, communication delays will not allow for remote tele-operation; instead, more capable robots are envisioned working side-by-side with astronauts.

This demonstration is limited to a single team composition, which consists of one extra-vehicular astronaut (i.e., outside the spacecraft, EV), a humanoid robot (e.g., Robonaut 2

[Diftler et al., 2011]), a remote manipulator system (RMS, e.g., Canadarm [Hiltz, Rice, Boyle, and Allison, 2001]), and a free-flying robot (e.g., Astrobee [Bualat, Barlow, Fong, Provencher, and Smith, 2015]). Thus, compared to the case study in chapter 5 with a four-astronaut team, three of these astronauts are replaced with robotic agents, potentially reducing the risk involved with astronauts operating outside the vehicle, freeing up three astronauts to perform other tasks, or allowing a reduction in the crew-size.

This case study will again evaluate the three work allocation presented in the case study in chapter 5, each striving for a different type of coherency, but are modified to re-allocate work from the astronauts to the robotic agents. Figure 6.2-Figure 6.4 show the work allocations in graphic form.

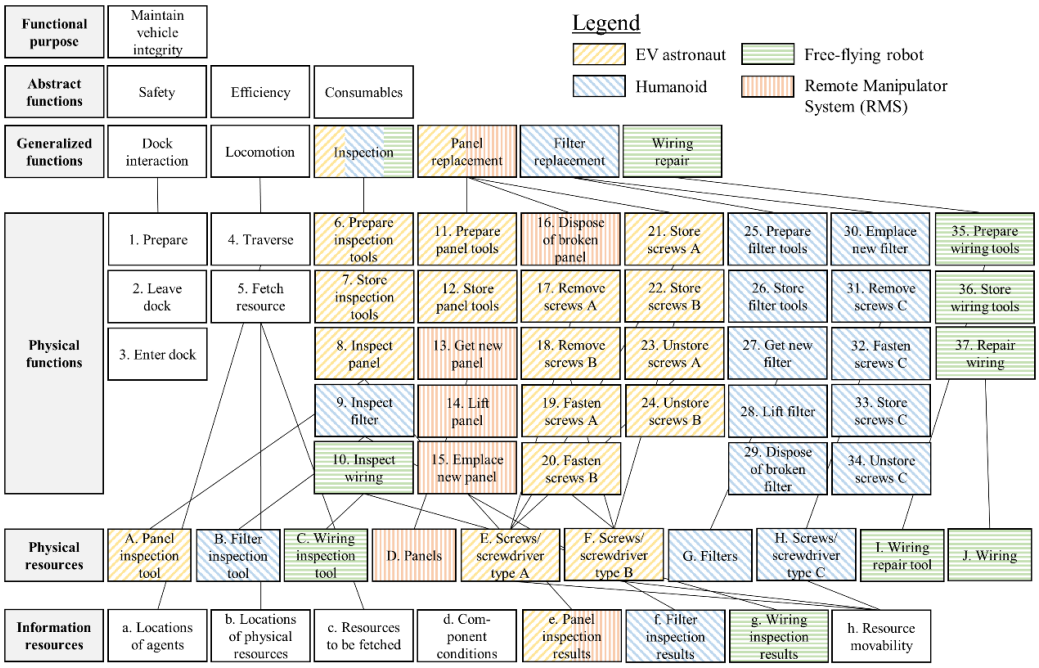


Figure 6.2: Work allocation 1, with coherency based on generalized functions.

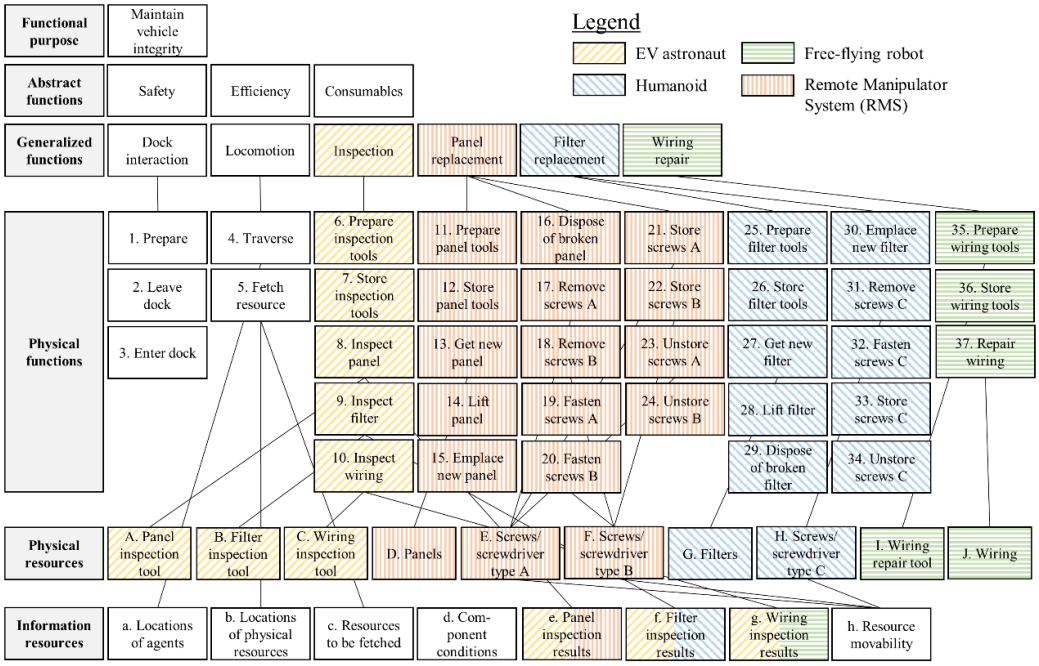


Figure 6.3: Work allocation 2, with coherency in physical and information resources, and some opportunities for parallel work.

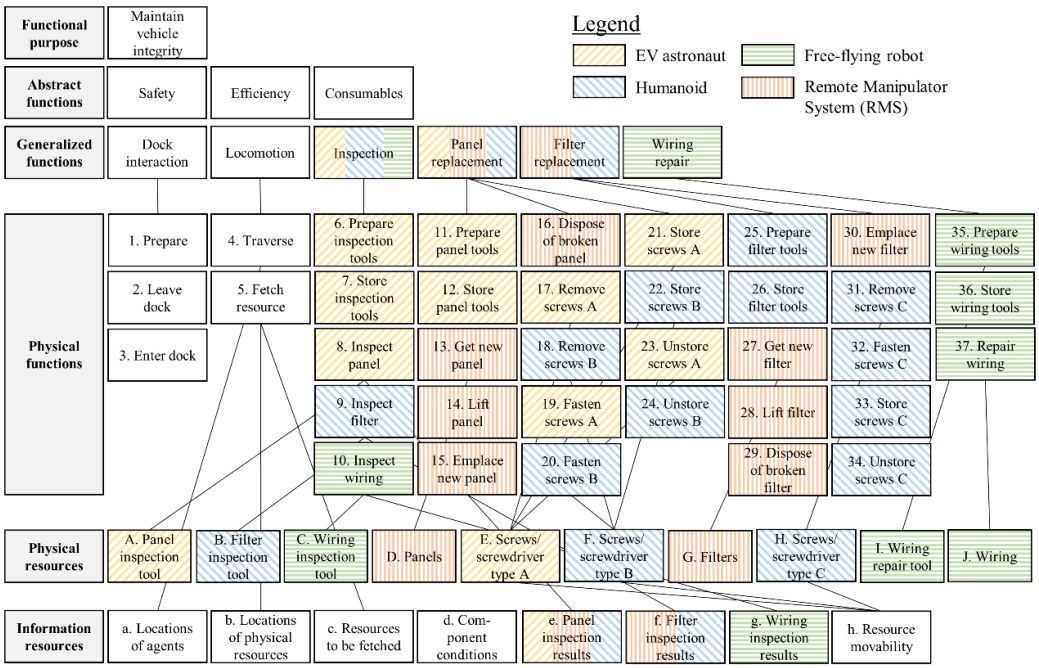


Figure 6.4: Work allocation 3, created to accommodate parallel work to reduce mission duration.

It is assumed that robots in this scenario cannot be held accountable for the outcome of an action, so human agents are allocated responsibility for the outcome of the robotic actions. Through this mismatch between authority and responsibility, there is thus a need for additional teamwork actions for the responsible human agent to verify the robotic operations, which can be either through real-time monitoring of the robot, or through confirmation of the robot's status after each robotic action. Additionally, the current state of space robotic technology requires a human operator to provide directions and input.

To take on some of the heavy teamwork requirements associated with the robotic team mates, an intra-vehicular astronaut (i.e., inside the spacecraft, IV) and a Mission Control Center (MCC) agent are added to the team to verify and control the robotic actions remotely. The allocation of responsibility in this case study has the EV astronaut responsible for the RMS's operation, the IV astronaut responsible for the Humanoid's operation, and MCC responsible for the free-flying robot; this pairing also identifies which humans are controlling each robot.

The human-robot interaction modes are limited by the available communication mechanisms between the agents. Both EV and IV astronauts have real-time communication links with the robots and can therefore execute the required teamwork synchronously and/or asynchronously with these robots. The MCC, however, is at a slight communication delay (10s), sufficiently large that the MCC can only work with the free-flying robot in an asynchronous manner, thus needing teamwork actions in the form of prior commanding and posterior confirmation. Furthermore, even though communication mechanisms are not a constraining factor for the EV and IVs interaction modes, it is assumed that only a limited number of interaction modes are supported by the RMS and Humanoid robots: the RMS

needs to be operated and monitored synchronously, and the humanoid robot can be controlled only through command sequencing.

With this work allocation and the constrained interaction modes, the teamwork can be added to the graph to identify its dependencies with the taskwork, or, when assuming predefined patterns of teamwork and its dependencies, can be automatically engendered based on simple design heuristics (useful when the analysis is aimed at quickly exploring different work allocations and their teamwork requirements). Here, the latter case is used, and the teamwork is therefore automatically engendered. Command actions are linked to their parent taskwork action through an automatically engendered "commands" resource. In addition, the command action is linked to get all information that are also input to the parent taskwork action. Likewise, confirmation actions are linked to their parent action through getting all information resources that the parent action sets. In addition, a "confirmation" resource is engendered that is set by the confirmation action and gotten by all taskwork actions that are also linked to the resources of parent taskwork.

The work model can then be simulated to asses is the dynamic effects of these additional teamwork actions and their dependencies with the taskwork. Like in the previous chapter, a taskload limit of one was assumed for all agents. However, contrary to the simulation of a human team, because the human-robot teamwork actions are considered more burdensome, the taskload effects of the teamwork is accounted for in the agent models. Thus, no agent can execute taskwork in parallel with teamwork actions.

Figure 6.5-Figure 6.7 show the timelines of actions for each agent with the three work allocations. The actions in grayscale are the taskwork action as originally identified in the

ADS. The actions in color (lower right of the legend) are automatically engendered teamwork actions, as recorded by the simulation engine based on the work allocation and feasible interaction modes. The timelines clearly show the impact of the heavy teamwork burden, its dependencies with the taskwork, and their effect on the work dynamics. For example, in all work allocations, taskwork actions for the EV astronaut need to be delayed because the EV astronaut also needs to tele-operate and monitor the RMS in handling the panels. Work allocation is most impacted by these effects, as more work allocated to the RMS poses a higher teamwork burden to the EV astronaut. A detailed inspection of the timeline additionally revealed that, because the EV astronaut is required to control and monitor the Humanoid robot while it is replacing a panel, the subsequent inspections are delayed, which results in long idle times for the other agents and a longer mission duration.

In addition, dependencies through teamwork for commanding and confirming robotic actions result in many interdependencies between the MCC and the free-flying robot, where these two agents need to continuously coordinate the timing of their work. The work dynamics thus result in the MCC intermittently needing to execute teamwork actions, with periods of high and low teamwork loads. These patterns are different for each work allocation, as determined by the temporal behavior of the rest of the team. Thus, these are examples of how the constraints and dependencies, in both the taskwork and teamwork, intertwine in emergent ways that would not be foreseeable from a static analysis.

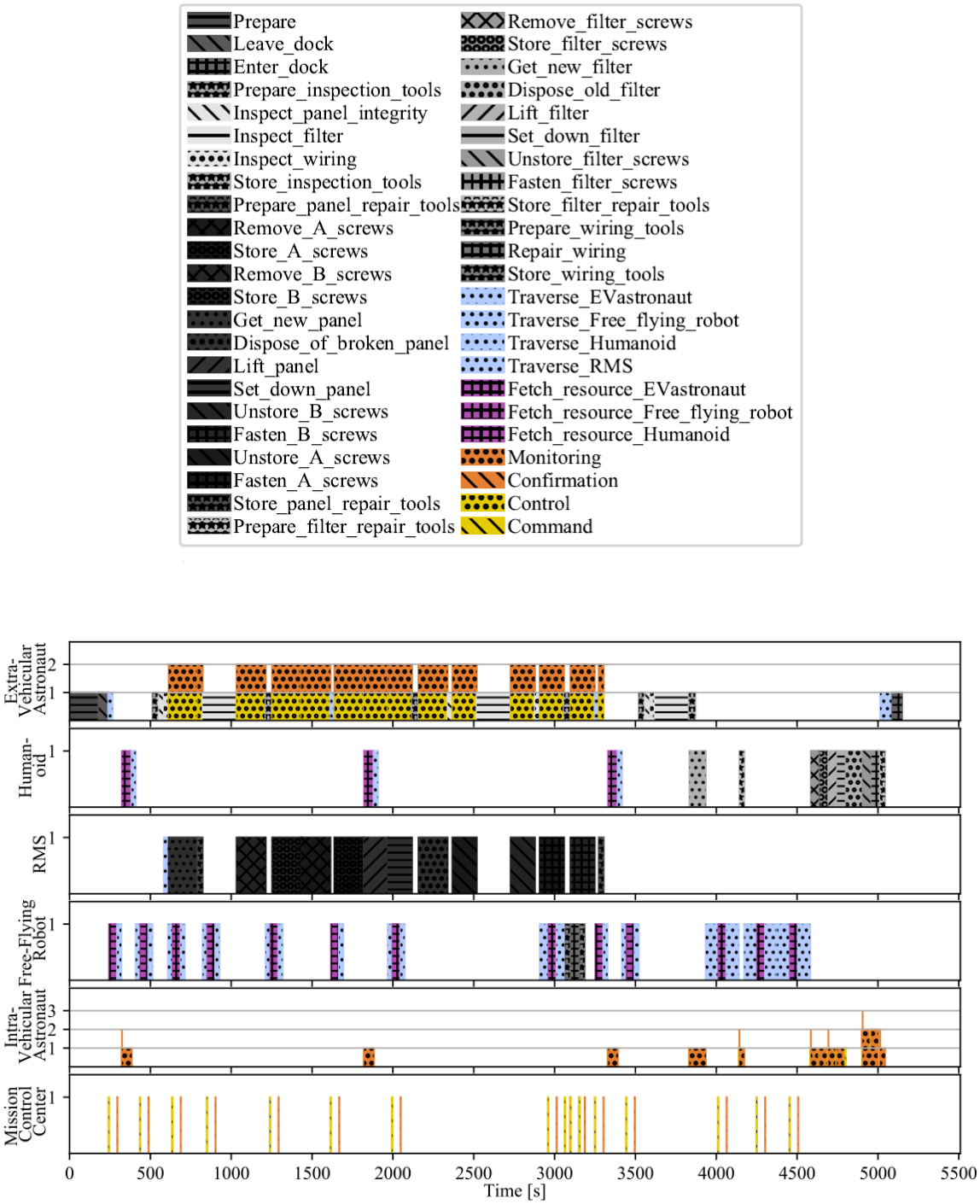


Figure 6.5: Timeline for work allocation 1.

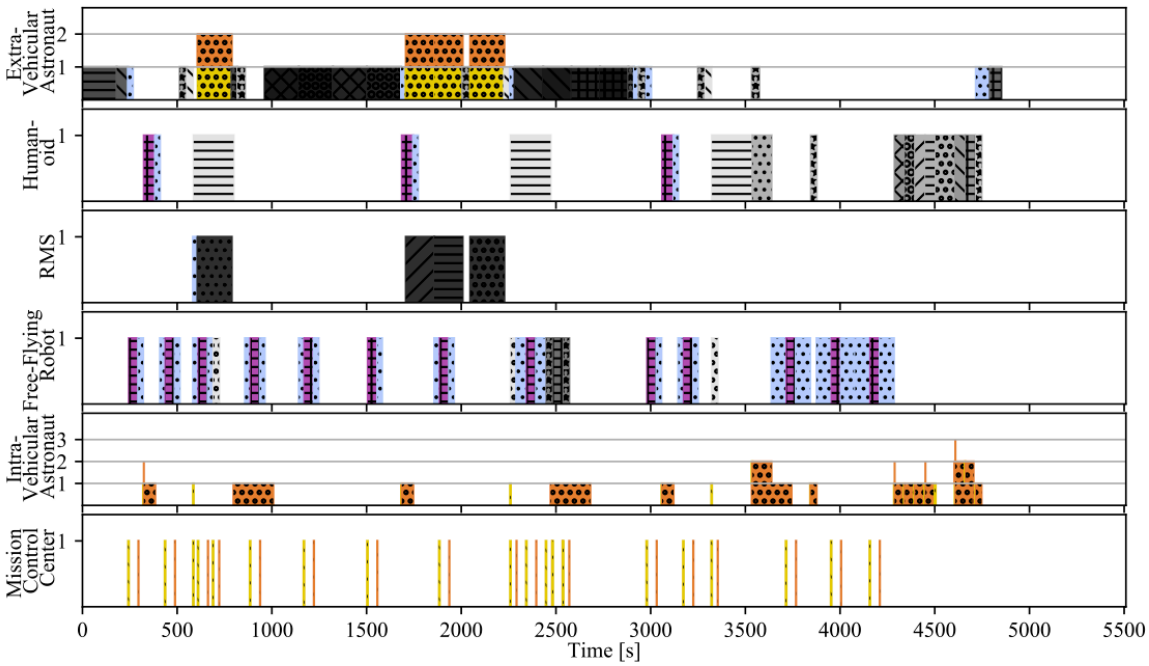


Figure 6.6: Timeline for work allocation 2.

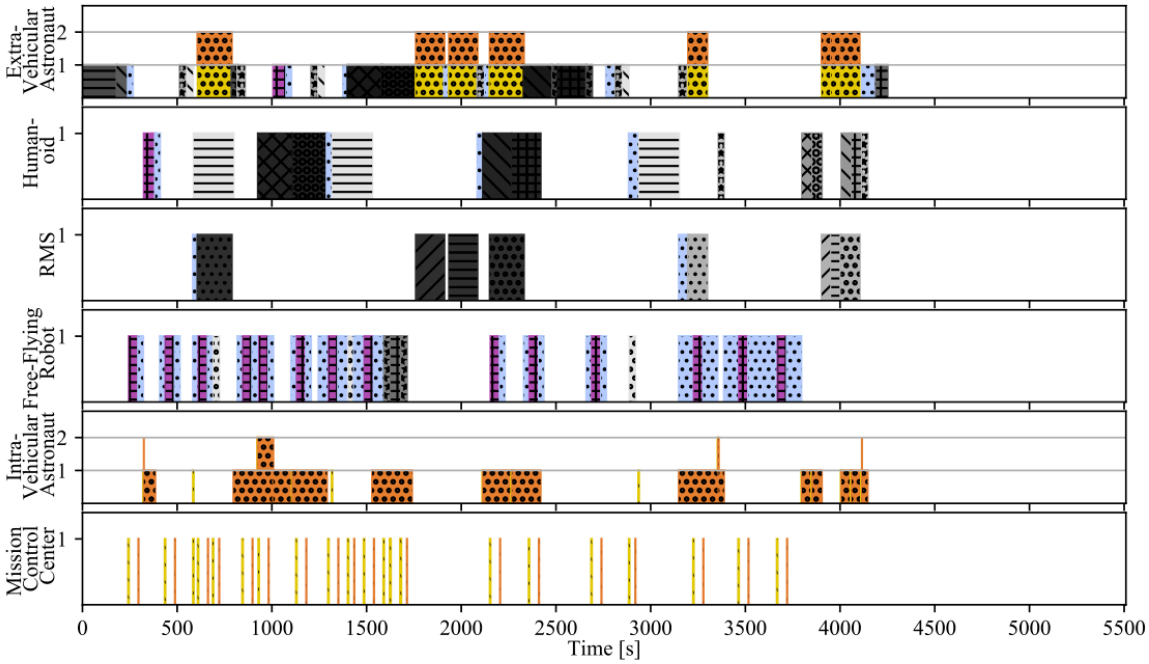
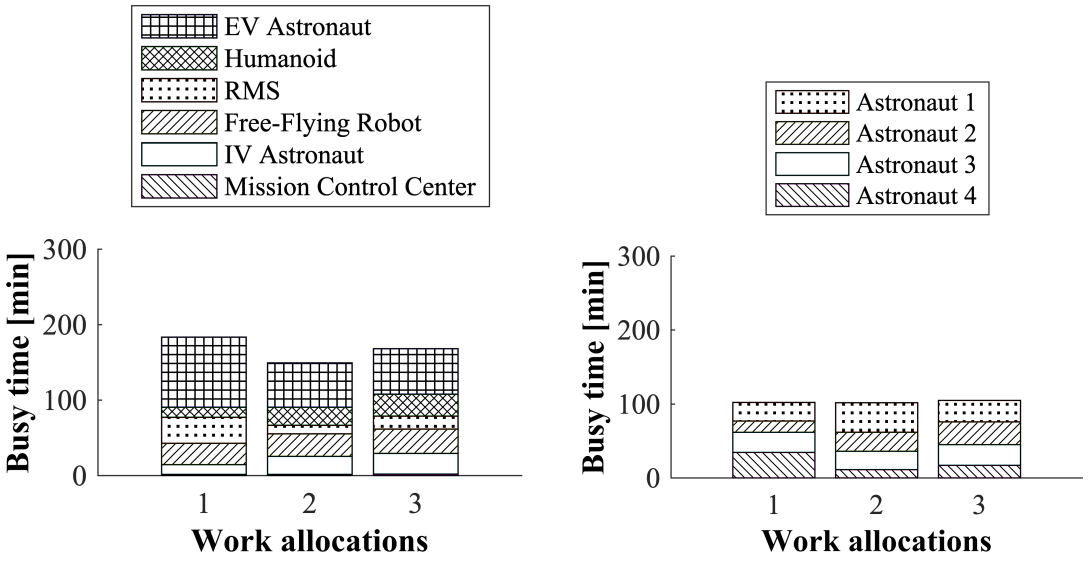


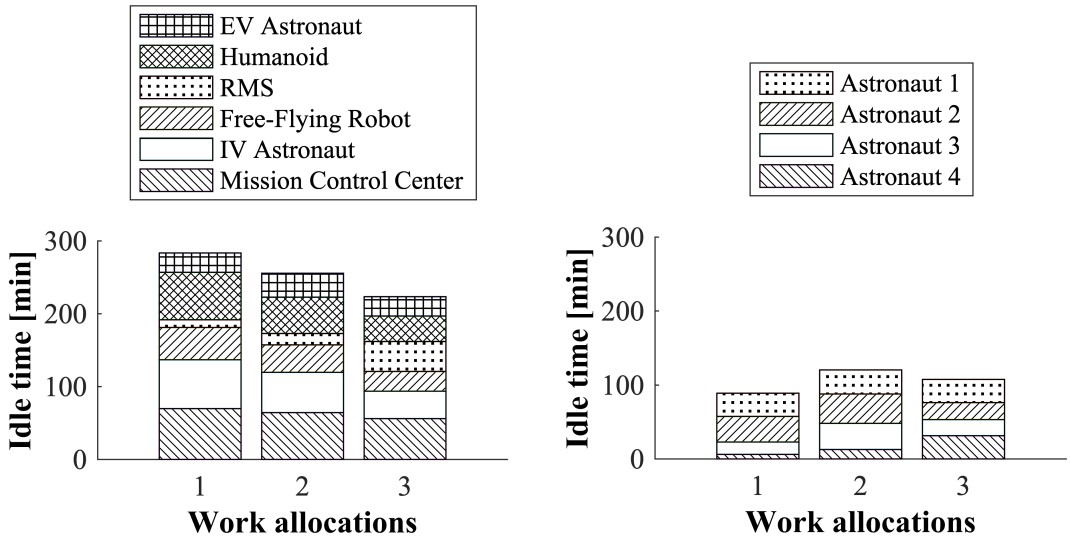
Figure 6.7: Timeline for work allocation 3.

Figure 6.8 and Figure 6.9 compare the work allocations in busy and idle time, respectively. The results from the human team in chapter 5 have been included in these figures to allow for a side-by-side comparison. The results clearly show that the effect of the overhead of teamwork, not just in terms of duration (reflected in the increased busy times) but more importantly in the idle time. Most of the increase in idle time can be led back to the inclusion of two additional agents, but even without the IV astronaut and MCC's contributions to idle time, the human-robot team shows significantly longer periods of idling. This demonstrates that the dependencies between the taskwork and the teamwork can have a major effect on the team's interweaving of actions, and highlights the importance of explicitly considering the effect of teamwork in a dynamic evaluation.



(a) Human-robot team (b) Human team

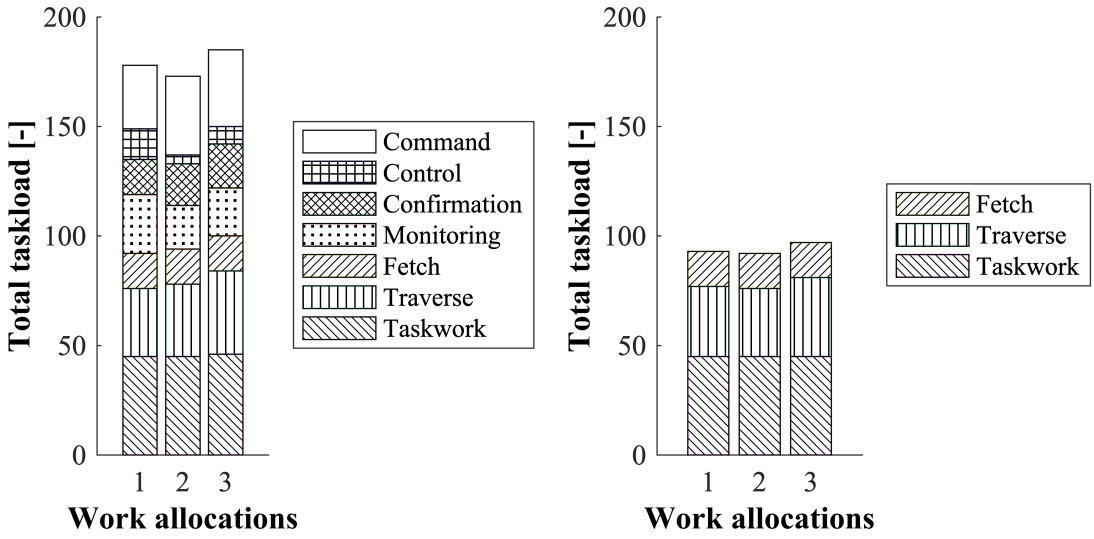
Figure 6.8: Comparison of the work allocation in busy time.



(a) Human-robot team (b) Human team

Figure 6.9: Comparison of work allocations in idle time.

Figure 6.10 shows the taskload for each work allocation, split between taskwork and different forms of teamwork. Work allocation 2 shows the lowest taskload, due to lower monitoring and control demands. Work allocation 3 still has the highest taskload, mostly because of a high fetching load, similar to the human team. Finally, Figure 6.11 shows the required exchange of information and transfer of resources. Compared to the human team (see Figure 5.11), the number of required information exchanges has nearly doubled, which can be attributed to the additional information exchanges necessary to conduct the teamwork. Moreover, patterns have reversed: whereas in the human team work allocation 1 had the highest need for information exchange (then due to its incoherency in information resources), here it has the lowest. Work allocation 2 (created for coherency in information and physical resources) still shows a comparatively low number of information exchanges. The required transfer of physical resources is unchanged.



(a) Human-robot team (b) Human team

Figure 6.10: Comparison of the work allocations in total taskload.

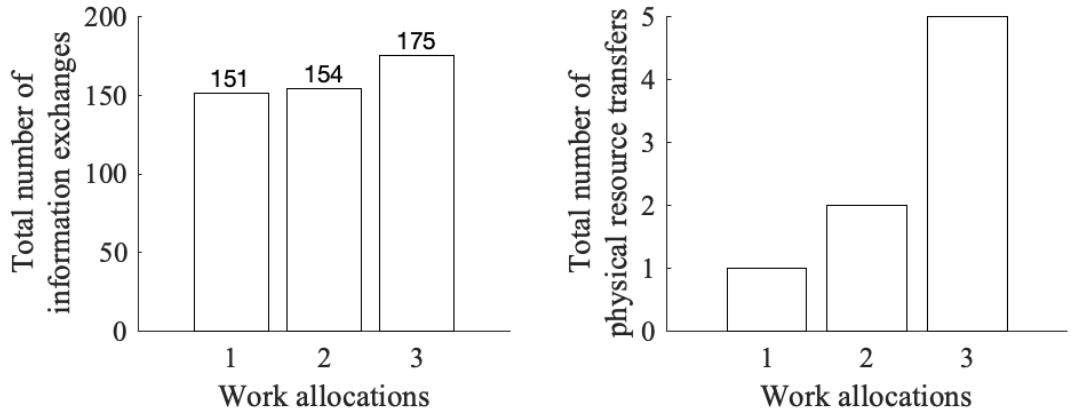


Figure 6.11: Required exchange of information and transfer of resources.

It should be noted that this case study may be assuming a fairly extreme distinction between human and human-robot teams, as it can be argued that the human team will also need some form of teamwork to coordinate (as shown in case study II of chapter 5). However, it is expected that many of the differences seen here will hold even when explicitly considering teamwork in the human team: humans are comparatively fluent in coordinating with other humans, therefore fewer teamwork actions will be necessary and their durations will be shorter. Additionally, fluency of humans may allow for dependencies between taskwork and teamwork being managed more proactively, reducing idle time.

6.4 Case Study II: Lunar Rover

The second case study will further highlight how rigidity of a robotic team member may limit work strategies. The case study from chapter 5 is continued but one astronaut team member is changed to a rover agent, representing the rover's automated capabilities. This has several implications for the work strategies and work allocation identified in earlier chapters. First, because the rover cannot be responsible for its own actions, team design 2 (chapter 5, in which responsibility was allocated to maximize cross-checking between the two astronauts) is no longer feasible. Thus, this limitation restricts the allocation of responsibility to the human agent. Table 6.1 shows one feasible allocation, in which the astronaut serves as a supervisor (or commander) to the rover agent. Similar to team design 1 in chapter 5, the astronaut is allocated authority for actions associated with driving the rover, while the rover agent is allocated authority for actions associated with imagery.

Table 6.1: Allocation of authority and responsibility between astronaut and rover. A = authority, R = responsibility

Taskwork	**Team design 1**	
	Astronaut	***Rover***
1. Check battery levels	A/R	
2. Check temperature	A/R	
3. Assess location and attitude	A/R	
4. Plan rover path	A/R	
5. Estimate size of objects	R	A
6. Localize obstacles	R	A
7. Select next waypoint	A/R	
8. Move rover		
9. Change camera angle	R	A
10. Capture imagery	R	A

Second, it is assumed that the rover has limited modes through which it can interact with the astronaut, in that verification must be performed through posterior confirmation which, given the limited communication channel/interface between astronaut and rover, prevents the astronaut from performing other tasks simultaneously. Third, compared to human agents, the rover may have more rigid and higher standards for QOS of information resources, specifically for the safety-critical actions ('Localize obstacles'). Therefore, some of the work strategies identified for in this work model (see chapter 4) may no longer be feasible in more information-constrained situations.

Figure 6.12 shows the timeline of actions for this rover-astronaut team. The astronaut now has significantly more confirmation actions. The confirmation actions have a duration of 10 seconds, during which the astronaut cannot attend to any other tasks. Likewise, the rover sometimes needs to wait for the astronaut to verify its actions before it can continue with its next actions. Furthermore, the limitations of the rover require the full re-evaluation of the obstacles at each iteration of the work model.

Thus, compared to the earlier timelines shown in chapters 3 and 4, this team has relatively high coordination burden and is constrained to a particular sequence of actions. This is again a fairly extreme example of how robotic limitations can limit the team's available works strategies. In this case, the astronaut is forced to conform his/her own work patterns to those rigidly imposed by the rover. Thus, when faced with changing work demands, this astronaut-rover team has limited flexibility in terms of matching the work demands with appropriate work strategies. When this work strategy does not fit the demands of the work environment, the astronaut might switch to work strategies that limit the involvement of the rover (if possible), or the system might fail to perform efficiently and safely.

A final note on this case study is that with the insight provided through this evaluation of the work dynamics, more flexible work allocation, interaction modes and work strategies can be created. For example, further analysis of this astronaut-rover team can look at different allocations of the safety-critical actions, potentially reducing the dynamics effects of the confirmation overhead. Or further analysis of the dependencies might identify opportunities for more flexibility in the rover's work strategies (e.g., QOS requirements can perhaps be relaxed under certain work allocations). Likewise, considering different interaction modes, the duration of teamwork may be reduced to limit its ripple effects on other taskwork and teamwork.

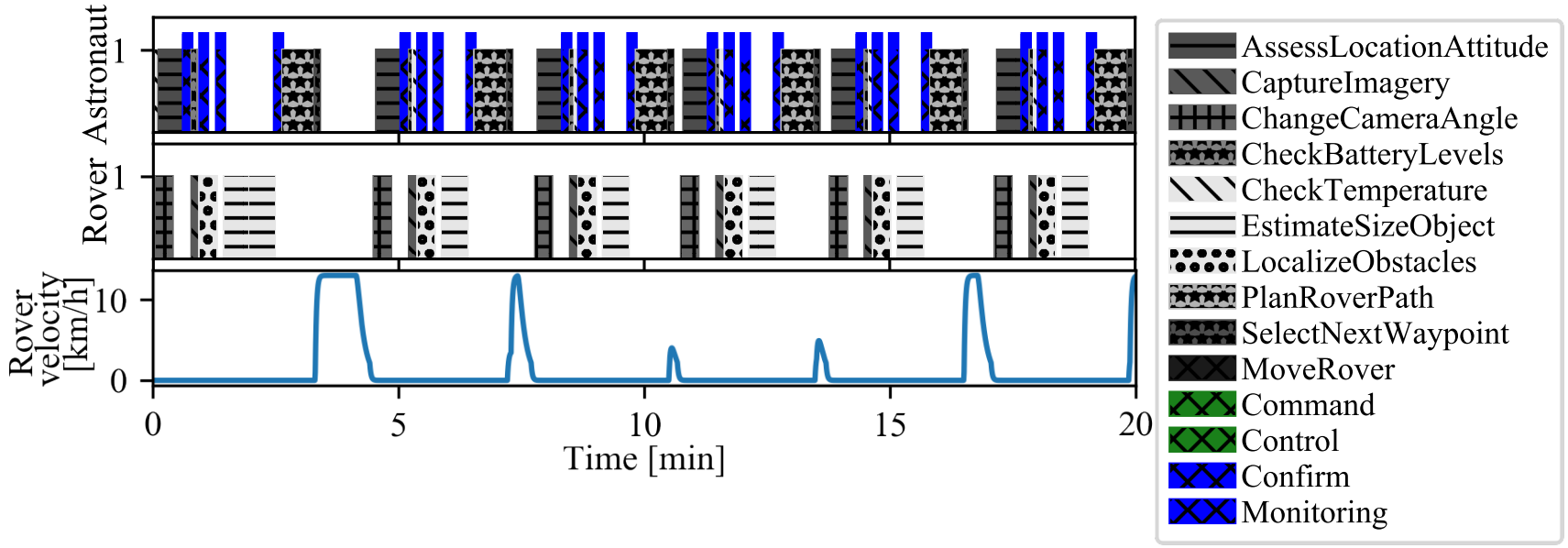


Figure 6.12: Timeline of actions in rover-astronaut team.

6.5 Discussion

Thus, a human-robot team's ability to adapt is determined, first, by the availability of feasible work strategies and, second, by how well characteristics of feasible work strategies match with the changing work demands. From this perspective, the analysis proposed here can be used to foster adaptation through design in a number of ways. First, an effective human-robot team is flexible when it supports multiple work strategies as identified through analysis of the graphs, ideally spanning a range of contextual factors and control modes (e.g., designers can evaluate candidate human-robot team designs for both opportunistic and strategic control modes, or different paths through the graphs). With the models presented in this thesis, one can purposefully design for dependent or independent work in a human-robot team, so that implications of dependencies can be accurately assessed and accounted for. Thus, one could analyze how human and robotic work processes are dependent through various levels of abstraction and can or cannot coexist, dependently or independently. For example, if a robotic team member is highly rigid, effective team design could allocate the critical path in the work to human team members only, so that the team as a whole still has the ability to adapt.

Second, an effective human-robot team is flexible in the interaction modes between humans and robots, again supporting multiple work strategies for coordinating the collective work of the team. Good design can strive for creating efficiency in human-robot interaction, thereby reduction duration of overhead teamwork actions. Using an evaluation of the graph and simulation of the work dynamics, one can identify how this teamwork overhead affects the rest of the team and identify specific situations or design conditions that would benefit most from an improvement in human-robot interaction.

The limitations of robotic team members and the implications thereof discussed in this chapter are broadly applicable to technology and unlikely to be resolved in the near future. Thus, it is imperative that in the design of human-robot teams these issues are accounted for from the earliest phases of design. As demonstrated in the case studies, computational models of work can model and predict several common issues in human-robot teams, including the significant overhead of human-robot interaction and teamwork, and their temporal effects. Good design can use these predictions to explore different strategies for allocating and coordinating work between human and robotic team members, such that human agents can finish-the-design as appropriate in the immediate context.

In conclusion, the approach presented here has several advantages over existing framework for human-technology integration. First, existing methods for human-robot or human-automation interaction design have a limited perspective in only considering agent capabilities (Fitts et al., 1951) or "levels" for decision-making (Parasuraman et al., 2000; Sheridan and Verplank, 1978) as a basis for design. The approach discussed in this chapter considers agent capabilities as one of many factors, but includes more intricate analysis of the consequences of limited capabilities in terms of teamwork requirements and their effects on the team's work strategies, in a manner similar to coactive design (Johnson, 2014). Second, the computational models of work then allow for a dynamic evaluation of a human-robot team's interaction and interweaving of activities, identifying any emergent behavior stemming from the complex relationships between the work, the work environment and the agents.

6.6 Summary

The method presented in this thesis can be used to account for common problems in human-robot teams, by allowing designers to gain insight in how their design choices alleviate or aggravate these issues. Robots are limited team members, which has several implications when applying the methods presented in this thesis to the design of human-robot teams. First, robots are rigid and clumsy in their execution of taskwork. Second, robots cannot be allocated responsibility, which implies authority-responsibility mismatches requiring a human supervisor. Third, robots are constrained in their interaction modes.

These limitations can be accounted for in the design framework present in this thesis. Robotic rigidity is reflected in the strategies analysis as a limited number of paths and topological orders that can be taken. Authority-responsibility mismatches and limitations in interaction modes can be accounted for in relatively stringent and constrained teamwork with durations.

CHAPTER 7. CONCLUSIONS

The focus of this thesis is on design of teams to foster the ability to adapt work strategies in complex work domains. Effective team design, from this perspective, supports flexibility in strategies for conducting the team's collective work, spanning a wide range of potential operating conditions. In addition, effective teaming requires strategies to support the dynamics of the work, with the work of team members fluently coordinated and interweaved. The modeling in this thesis aims to create a systematic basis for inquiry into potential design issues related to this perspective of effective teaming. The contributions of this thesis are summarized in section 7.1, and limitations and future work are discussed in section 7.2.

7.1 Summary of Contributions

The first contribution of this thesis is the computational models of a team's collective work, including the use of tools, the sharing of information and the interaction between team members. The computational work models address the critique that existing methods for team design are static and qualitative– they evaluate design options based on a general assessment on paper, perhaps based on prior experience or subject-matter expertise – and therefore do not take into account the temporal behavior and interaction that is emergent in a multi-agent team until later-in-design human-in-the-loop tests.

Indeed, the computational models provide several benefits over traditional methods. First and foremost, through simulation of the work models, the dynamic and emergent nature of the work of teams can be analyzed directly. This dynamic evaluation of a team's

work provides the ability to evaluate how well a team design supports the dynamics of the work and can identify emergent effects that would not be apparent from a static analysis alone. Particularly early in design, when major design decisions are being made related to work allocation and interaction modes, these predictions can help identify and prevent issues that would otherwise be discovered during more advanced design phases (e.g., through field testing or human-in-the-loop experiments), when design changes are costly.

Second, the simulation of a computational work model can help verify that a work model is reasonably complete, through checking the degree that expected behavior or manipulation of the work environment is reflected in the simulation results. The modeling of work in computational form and the explicit accounting of how the work is dependent on the work environment and vice versa helps identify missing or incorrect elements and dependencies in an ADS. Thus, the dynamic evaluation provides a model verification not present in the traditionally static analysis of an ADS.

The second contribution of this thesis is a systematic approach for identifying and evaluating work strategies in multi-agent teams, accounting for dependencies between the work and the team's work environment. This contribution addresses the need for systematic and formative methods for designing teams. Earlier methods, including CWA, are conceptual in their nature and lack the systematicity desired by practitioners not well-versed in the underlying material. The approach laid out in this thesis takes a similar holistic approach, and uses ideas first discussed in the CWA framework, but contributes more systematic modeling constructs (i.e., the graph representation of an ADS, methods from graph theory, and the simulation of work strategies) that allow for a more structured, broad-sweeping, and efficient analysis of the design trade-space.

The third contribution is a method that can explicitly account for the required coordination and teamwork resulting from interdependencies between agents. The graph representation and the associated consideration of work clusters provide formative insight into how dependencies in the work drive the team's requirements for coordination. Then, with simulation of the computational work models, the coordination requirements can be explicitly accounted for through the exchange of information resources and transfers of physical resources. Furthermore, the engendering of teamwork actions associated with coordination requirements (i.e. from first principles, the minimum necessary teamwork) contribute evaluation of the overhead associated with coordination, in terms of the additional taskload and the additional dependencies it creates given different interaction modes. The graph analysis and subsequent simulation can help designers predict how teamwork impacts the feasible work strategies for the team.

Together, these three contributions address the critique that current design methods for human-robot teams do not provide a comprehensive analysis of how robotic limitations may affect a team's ability to adapt. The systematic approach presented in this thesis can be used to explicitly consider the broader implications of the limitations of robotic team members on effective team design: It provides means to evaluate the effectiveness of a conceptual team design by identifying the flexibility of a human-robot team design in terms of the set of feasible strategies, and, through simulation of computational work models, evaluating how well a team's feasible work strategies support the dynamics of the work, including the required coordination and associated teamwork.

Finally, the graph analysis and computational work models have the added benefit that they scale easily and are efficient in time and labor, allowing broad applicability of the

approach from quick "what-if" experiments to the full mapping of the design trade-space for a team. The simulation and analysis of work clusters are scalable without issues – there is an initial effort involved in creating the computational models of work, but once these are established their simulation is computationally inexpensive. Identification of work strategies through graph analysis can be scaled by employing methods from graph theory to analyze the paths in an efficient manner. In addition, the graph analysis poses an efficient pre-analysis to the more detailed evaluation through simulation. Thus, in very large-scale work models, a bounded set of work strategies and work allocations of interest may be identified through analysis of the graph representation, which can then be simulated to identify emergent work dynamics.

7.2 Limitations and Directions for Future Research

The approach for human-robot team design presented in this thesis has several limitations, some of which provide directions for future research. First, it is important to recognize that although simulation can account for many important dynamics of work, it does not capture all dynamics that are going on in a team. The simulation assumes perfect execution of the work with immediate transitions between actions, identifying the work dynamics that even a perfect team would need to abide by. Thus, it is very suitable for identifying the minimum requirements for design. However, other dynamics not captured by the simulation might also play a role in a team's performance. For example, a recent experiment in which data from WMC was compared to human-in-the-loop data indicated that there is a variable cost involved in switching from one action to the other, especially when coordinating through teamwork (Ma, Ye, IJtsma, Pritchett, and Feigh, 2019). In addition, the simulation requires estimation of action durations. Thus, the results represent

an estimate of what will happen given a best possible prediction of action durations. These predictions can be obtained from historical data or simple experiments but may turn out to be different once the system is fielded.

The computational models of work can be extended to capture variation and uncertainty in the agents' execution of work. Such variation can be modeled in several ways, but a relatively easy way is to vary action durations, which could include both taskwork and teamwork. For example, action durations can be selected based on contextual factors or from a random distribution. Alternatively, performance variations can be captured in the output of actions, where, depending on contexts or based on random variables, the manipulations of resources might differ. Multiple runs of the simulation can then assess how the work dynamics are affected and calculate a distribution of metrics. Of note here is that performance modeling in light of the computational work models is most useful when done through a set of rules that are reflective of characteristics of work and the work environment. There is a long history of human performance modeling to identify "internal" processes of agents. However, the assumptions implied in such modeling quickly lead back to normative or simplified models, away from the analysis of inherent characteristics of the work and work domain.

Second, a limitation of the graph analysis presented in chapter 4 focused primarily on the lower half of the ADS, modeling how the physical representation and characteristics of a team's work affect the work strategies available to the team. The higher-level consideration were considered as summations of work allocations in chapters 5 and 6, but future work could expand on how higher-level considerations associated with the values and priorities of the team relate to different work strategies, including how the allocation

of work, creating coherency or incoherency at the higher levels, affects the managing of values and priorities as the team adapts. For example, Jennins (1996), focusing on distributed artificial intelligence, proposed analysis of coordination through the modeling of a system's goal structure and the identification of interdependencies between agents sharing goals and resources. Thus, whereas this thesis has mostly analyzed how emergence flows up the abstraction hierarchy (as dependencies with resources that determine dynamics of work), future work can explore how emergence might flow down the hierarchy.

Related to this, when originally introducing the ADS, Rasmussen, Pejtersen, and Goodstein (1994) noted that systems can be classified on a continuum between systems whose behavior is governed by natural laws (e.g., first principles of physics) and systems whose behavior is solely governed by the agent's intentions. In between are systems whose behavior needs to abide by nature's laws but are also governed by actor's intentions. Where emergence up and down the levels of abstraction meet (and how trade-offs may need to be made as teams adapt) is an interesting intersection for further analysis of work strategies.

Third, one factor in a team's work dynamics not captured in the current modeling is a team's organizational structure, as hierarchical or flat relationships between all team members. The modeling of teamwork in this thesis has focused primarily on dyads of agents involved in joint work, with observability and directability requirements between two agents. Beyond dyads, a team's organizational structure can result in these requirements spanning multiple levels, with associated need for teamwork. For example, in space operations, the mission control center might be ultimately responsible for all operations, and verification of action outcomes and action input might therefore need to go

through several hierarchical levels (e.g., robot to astronaut supervisor, to astronaut commander, to mission control, and vice versa).

Thus, future research could extend computational models of work to analyze teamwork requirements under different team hierarchies, for example by 'stacking' the teamwork: with an astronaut monitoring a robot, the mission control center might in turn be required to monitor the astronaut (i.e., the mission control center is responsible for the astronaut's teamwork), with the astronaut relaying any relevant operational data about the robot. Likewise, the mission control center might provide commands to an astronaut, who in turn relays those commands to a robotic agent. There are design questions related to organizational structures that such modeling extensions could aim to contribute to, for example, to what degree does a team design constrain or allow work strategies to be to selected centrally versus locally? The modeling of different types and degrees of coherency in a team's collective work, together with modeling of a team's organizational structure, could help evaluate how local changes to an agent's actions have ripple effects on the global team.

Fourth, although this thesis provided some notion of how different work strategies can be reflective of different contextual control modes, future work can further elaborate on supporting multiple contextual control modes in design through computational modeling of work. Such efforts can also contribute to the questions as to how individual team member's control modes would relate to each other in a team setting, and how these might affect a 'team control mode'. While Stanton, Ashleigh, Roberts, and Xu (2001) showed that a team's behavior can be classified according to different control modes, certain work conditions might warrant individual team members or sub-teams operating in different

control modes. For example, if the work demands are high for only one particular team member, warranting more opportunistic control modes, another team member might assist by proactively managing any dependencies with the overloaded team member. Thus, rather than defining a work strategy for the entire team, such analysis could focus on individual strategies and how the work would be coordinated in such scenarios.

Finally, the case studies presented in this thesis were meant as demonstration of the concepts, and further research is needed to apply the concepts of this thesis to study common issues or challenges encountered in practice. The replication of real-life teaming problems can serve to further validate the approach as being able to accurately predict and/or analyze relevant issues. Where available, historical operational data of human-robot teams can be used to assess how well the simulation is able to predict actual operational issues. Alternatively, data can be obtained from human-in-the-loop studies. In a recent experiment, data from WMC was compared to data obtained from a human-in-the-loop study, which validated that the simulation can reasonably well predict patterns in mission duration, idle and busy times under different work allocations and interaction modes (Ma et al., 2019).

Beyond validation with historical or human-in-the-loop data, application during a team design process can evaluate how well the approach leads to usable insight during design. In particular, such analysis can analyze how the insight from this approach can be translated into specific design requirements for a team's architecture, for example, interfaces or robotic capabilities or algorithms, and how the approach can be complemented by, or used in conjunction with, other methods for team design. Most notably, coactive design has proven a powerful method for designing resilient human-robot collaboration, through the

explicit identification and support of interdependence relationships (Johnson, 2014). Coactive design identifies alternative paths through decomposed elements of a human-machine pair's work, modeling how the human can support the machine and vice versa and identifying OPD requirements associated with potential interdependencies between the agents. Thus, whereas this thesis focused on the dependencies between the work and the work environment, and how these dependencies need to be managed temporally in a multi-agent team, coactive design explicitly analyzes dependencies related to each team members capabilities, and their implications for interface and machine algorithm design (Johnson et al., 2019). Thus, in potential each method provides complementary insight, where a dynamic analysis can help identify and/or verify a team's work model, including its dependencies and interaction with the work domain, and coactive design can identify deeper interdependency relationships between agents, beyond the elemental teamwork actions considered in the computational work models, including their implications for interface and algorithm design.

REFERENCES

Adams, M., Hall, W., Hanson, M., Zacharias, G., McEneaney, W., & Fitzpatrick, B. (2002). Mixed Initiative Planning and Control Under Uncertainty. In *AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*. Portsmouth, VA.

Ahlstrom, U. (2005). Work domain analysis for air traffic controller weather displays. *Journal of Safety Research*, *36*, 159–169.

Akin, D. L., Hedgecock, J. C., & Sorenson, E. A. (1989). Experimental Results of Integrated EVA/Telerobotic Work Sites. In *NASA Space Operations and Robotics Workshop*.

Armstrong-Wright, A. T. (1969). *Critical path method: Introduction and practice*. Prentice Hall Press, UK.

Army Research Labratory. (2009). *IMPRINT Pro User Guide* (Vol. 2).

Ashoori, M., & Burns, C. (2013). Team cognitive work analysis: Structure and control tasks. *Journal of Cognitive Engineering and Decision Making*, *7*, 123–140.

Bhattacharyya, R. P., & Pritchett, A. R. (2017). Designing function allocations in air traffic concepts of operation using network optimization. *Journal of Air Transportation*, *25*, 61–71.

Błazewicz, J., Domschke, W., & Pesch, E. (1996). The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, *93*, 1–33.

Bodin, I., & Krupenia, S. S. (2016). Activity Prioritization to Focus the Control Task Analysis. *Journal of Cognitive Engineering and Decision Making*, *10*, 91–104.

Boeke, D., Miller, M., Rusnock, C., & Borghetti, B. J. (2015). Exploring Individualized Objective Workload Prediction with Feedback for Adaptive Automation. In *Proceedings of the 2015 Industrial and Systems Engineering Research Conference* (pp. 1437–1446).

Bradshaw, J. M., Hoffman, R. R., Woods, D. D., & Johnson, M. (2013). The Seven Deadly Myths of "Autonomous Systems." *IEEE Intelligent Systems*, *28*, 54–61.

Bualat, M., Barlow, J., Fong, T., Provencher, C., & Smith, T. (2015). Astrobee : Developing a Free Flying Robot for the International Space Station. In *AIAA SPACE 2015 Conference and Exposition*.

Burkhalter, B. B., & Sharpe, M. R. (1995). Lunar Rover Vehicle: Historical Origins,

Development and Deployment. *Journal of British Interplanetary Society*, *48*, 199–2.

Christoffersen, K., & Woods, D. (2002). How to Make Automated Systems Team Players. *Advances in Human Performance and Cognitive Engineering Research*, *2*, 1–13.

Christopher, M. (1992). *Logistics & supply chain management*. *Logistics and Supply Chain Management: Creating Value-adding Networks*.

Colombi, J. M., Miller, M. E., Schneider, M., McGrogan, J., Long, D. S., & Plaga, J. (2012). Predictive Mental Workload Modeling for Semiautonomous System Design: Implications for Systems of Systems. *Systems Engineering*, *15*, 448–460.

Costes, N. C., Farmer, J. E., & George, E. B. (1972). *Mobility Performance of the Lunar Roving Vehicle: Terrestrial Studies, Apollo 15 Results*. *NASA Technical Report*. Washington, DC.

Cquisti, A. A., Sierhuis, M., Clancev, W. J., & Bradshaw, J. M. (2002). Onboard the International Space Station. In *Proceedings of the 11th Conference on Computer-Generated Forces and Behavior Representation*. Orlando, FL.

Deans, M., Fong, T., Lee, P., Hodges, K., Helper, M., Landis, R., … Kobayashi, L. (2009). Robotic scouting for human exploration. In *AIAA Space 2009 Conference & Exposition*. Pasadena, USA.

Defense Science Board. (2012). *The Role of Autonomy in DoD Systems*. Washington, DC. https://doi.org/10.1016/S0140-6736(02)11924-6

Demira, M., McNeeseb, N. J., & Cookea, N. J. (2018). Dyadic Team Interaction and Shared Cognition to Inform Human-Robot Teaming. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *62*, 124–124.

Diftler, M. A., Mehling, J. S., Abdallah, M. E., Radford, N. A., Bridgwater, L. B., Sanders, A. M., … Ambrose, R. O. (2011). Robonaut 2 - The first humanoid robot in space. *Proceedings - IEEE International Conference on Robotics and Automation*, *1*, 2178–2183.

Diggelen, J. Van, Bradshaw, J. M., Grant, T., Johnson, M., & Neerincx, M. (2009). Policy-based design of human-machine collaboration in manned space missions. In *Proceedings of the Third IEEE International Conference on Space Mission Challenges for Information Technology* (pp. 376–383). Pasadena, USA: IEEE Computer Society Washington.

Dijkstra, E. W. (1959). A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, *1*, 269–271.

Drake, B. G. (2009). *Human Exploration of Mars*. Houston, Texas.

Elfes, A., Weisbin, C. R., Hua, H., Smith, J. H., Mrozinski, J., & Shelton, K. (2008). The

HURON task allocation and scheduling system: Planning human and robot activities for lunar missions. *2008 World Automation Congress*, 441–448.

Entin, E. E., & Serfaty, D. (1999). Adaptive Team Coordination. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *41*, 312–325.

Feigh, K. M., & Pritchett, A. R. (2005). Airline Operations Managers: An Introduction to the Third Leg of the National Air Transportation System. In *6th USA/Europe ATM R&D Seminar*.

Feigh, K. M., & Pritchett, A. R. (2014). Requirements for effective function allocation: A critical review. *Journal of Cognitive Engineering and Decision Making*, *8*, 23–32.

Fischer, U., & Mosier, K. (2014). The impact of communication delay and medium on team performance and communication in distributed teams. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 58, pp. 115–119). Chicago, USA: SAGE Publications.

Fitts, P. M., Viteles, M. S., Barr, N. L., Brimhall, D. R., Finch, G., Gardner, E., … Stevens, S. S. (1951). *Human Engineering for an Effective Air-Navigation and Traffic-Control System*.

Fong, T., Abercromby, A., Bualat, M. G., Deans, M. C., Hodges, K. V., Hurtado, J. M., … Schreckenghost, D. (2010). Assessment of robotic recon for human exploration of the Moon. *Acta Astronautica*, *67*, 1176–1188.

Fong, T., Bualat, M., Deans, M., Allan, M., Bouyssounouse, X., Broxton, M., … Schreckenghost, D. (2008). Field Testing of Utility Robots for Lunar Surface Operations. In *AIAA SPACE 2008 Conference & Exposition*. San Diego, CA.

Fong, T., Zumbado, J. R., Currie, N., Mishkin, A., & Akin, D. L. (2013). Space telerobotics: Unique challenges to human-robot collaboration in space. *Reviews of Human Factors and Ergonomics*, *9*, 6–56.

Gauthereau, V., & Hollnagel, E. (2005). Planning, Control, and Adaptation: A Case Study. *European Management Journal*, *23*, 118–131.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems, Science and Cybernetics*, *4*, 100–107.

Helmreich, R. L., Merritt, A. C., & Wilhelm, J. A. (1999). The Evolution of Crew Resource Management Training in Commercial Aviation. *The International Journal of Aviation Psychology*, *9*, 19–32.

Hiltz, M., Rice, C., Boyle, K., & Allison, R. (2001). Canadarm: 20 years of mission success through adaptation. In *International Symposium on Artificial Intelligence*. Montreal, Canada.

Hoffman, G. (2013). Evaluating Fluency in Human-Robot Collaboration. *HRI Workshop on Human Robot Collaboration, 2013.*

Hollnagel, E. (1993). Contextual Control Models of Cognition. *Human Reliability Analysis: Context and Control*, 315–326.

Hollnagel, E. (2011). RAG-The resilience analysis grid. In *Resilience engineering in practice: a guidebook* (pp. 275–296). Farnham, Surrey: Ashgate Publishing Limited.

Hollnagel, E., & Bye, A. (2000). Principles for Modelling Function Allocation. *International Journal of Human-Computer Studies*, *52*, 253–265.

Hollnagel, E., Wears, R. L., & Braithwaite, J. (2015). *From Safety-I to Safety-II: A White Paper Professor*.

Hooey, B. L., Toy, J. J. N. T., Carvalho, R. E., Fong, T., & Gore, B. F. (2017). Modeling Operator Workload for the Resource Prospector Lunar Rover Mission. In *Advances in Human Factors in Simulation and Modeling: Proceedings of the AHFE 2017 International Conference on Human Factors in Simulation and Modeling* (Vol. 591, pp. 183–194). Los Angeles, CA.

Horneck, G., Facius, R., Reichert, M., Rettberg, P., Seboldt, W., Manzey, D., … Gerzer, R. (2003). *Study on the Survivability and Adaptation of Human to Long-Duration Interplanetary and Planetary Environments*. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/14696589

Hutchins, E. (1995). How a cockpit remember its speeds. *Cognitive Science*, *19*, 265–288.

International Space Exploration Coordination Group. (2018). *The Global Exploration Roadmap*. Washington, DC.

Johnson, M. (2014). *Coactive Design - Designing Support for Interdependence in Human-Robot Teamwork*. Delft University of Technology.

Johnson, M., Bradshaw, J. M., & Feltovich, P. J. (2019). Tomorrow's Human–Machine Design Tools: From Levels of Automation to Interdependencies. *Journal of Cognitive Engineering and Decision Making*, *12*, 77–82.

Johnson, M., Bradshaw, J. M., Feltovich, P. J., Jonker, C. M., Van Riemsdijk, M. B., & Sierhuis, M. (2014). Coactive Design: Designing Support for Interdependence in Joint Activity. *Journal of Human-Robot Interaction*, *3*, 43.

Kilgore, R. M., St-Cyr, O., & Jamieson, G. A. (2008). From Work Domain to Worker Competencies: A Five-Phase CWA. In A. M. Bisantz & C. M. Burns (Eds.), *Applications of Cognitive Work Analysis* (pp. 15–48). Boca Raton, FL: CRC Press.

Kirlik, A. (1993). Modeling strategic behavior in human-automation interaction: Why an "aid" can (and should) go unused. *Human Factors*, *35*, 221–242.

Klein, G., Woods, D. D., Bradshaw, J. M., Hoffman, R. R., & Feltovich, P. J. (2004). Ten Challenges for Making Automation a Team Player in Joint Human-Agent Activity. *IEEE Intelligent Systems*, *19*, 91–95.

Klein, Gary, Feltovich, P. J., Bradshaw, J. M., & Woods, D. D. (2004). Common Ground and Coordination in Joint Activity. In *Organizational simulation*. New York: Wiley.

Laughery, R. (1999). Using Discrete-Event Simulation to Model Human Performance in Complex Systems. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, & G. W. Evans (Eds.), *Proceedings of the 1999 Winter Simulation Conference* (pp. 815–820).

Looper, C. A., & Ney, Z. A. (2005). Extravehicular Activity Task Work Efficiency. In *35th International Conference on Environmental Systems*. Rome: SAE International.

Looper, C. A., & Ney, Z. A. (2006). Quantifying EVA Task Efficiency. *SpaceOps 2006 Conference*, 1–12.

Ma, L., Ye, S., IJtsma, M., Pritchett, A., & Feigh, K. (2019). An Experimental Refinement of Computational Models of Human-Robot Teams. In *AIAA Science and Technology Forum and Exposition 2020*.

Miller, M. J., McGuire, K. M., & Feigh, K. M. (2017). Decision Support System Requirements Definition for Human Extravehicular Activity Based on Cognitive Work Analysis. *Journal of Cognitive Engineering and Decision Making*, *11*, 136–165.

Moray, N., Sanderson, P. M., & Vicente, K. J. (1992). Cognitive task analysis of a complex work domain: a case study. *Reliability Engineering and System Safety*, *36*, 207–216.

Naikar, N. (2006). Beyond interface design: Further applications of cognitive work analysis. *International Journal of Industrial Ergonomics*, *36*, 423–438.

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A Model for Types and Levels of Human Interaction with Automation. *IEEE Transactions on Systems, Man, and Cybernetics*, *30*, 286–297.

Pomranky, R. A. (2006). *Human Robotics Interaction Army Technology Objective Raven Small Unmanned Aerial Vehicle Task Analysis and Modeling*.

Pritchett, A. R., & Bhattacharyya, R. P. (2016). Modeling the monitoring inherent within aviation function allocations. In *Proceedings of the International Conference on Human-Computer Interaction in Aerospace*. Paris, France: ACM New York.

Pritchett, A. R., Bhattacharyya, R. P., & IJtsma, M. (2016). Computational Assessment of Authority and Responsibility in Air Traffic Concepts of Operation. *Journal of Air Transportation*, *24*, 93–102.

Pritchett, A. R., Feigh, K. M., Kim, S. Y., & Kannan, S. K. (2014). Work Models that

Compute to describe multiagent concepts of operation: Part 1. *Journal of Aerospace Information Systems*, *11*, 610–622.

Pritchett, A. R., So Young Kim, Kannan, S. K., & Feigh, K. (2011). Simulating situated work. In *2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)* (pp. 66–73). IEEE.

Rasmussen, J. (1981). Models of Mental Strategies in Process Plant Diagnosis. In *Human Detection and Diagnosis of System Failures* (pp. 241–258). Boston, MA: Springer US.

Rasmussen, J., Pejtersen, A. M., & Goodstein, L. P. (1994). *Cognitive Systems Engineering* (1st ed.).

Rasmussen, J., & Vicente, K. J. (1990). The Ecology of Human-Machine Systems 11: Mediating "Direct Perception" in Complex Work Domains. *Ecological Psychology*, *2*, 207–249.

Salotti, J. M. (2011). 2-4-2 Concept for Manned Missions to Mars. In *62nd International Astronautical Congress*. Cape Town, South Africa: International Astronautical Federation.

Shah, J. A., Saleh, J. H., & Hoffman, J. A. (2007). Review and synthesis of considerations in architecting heterogeneous teams of humans and robots for optimal space exploration. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *37*, 779–793.

Sheridan, T. B., & Verplank, W. L. (1978). *Human and Computer Control of Undersea Teleoperators*. *Office of Naval Research*. https://doi.org/10.1016/S0262-4079(08)63018-3

Sierhuis, M., Bradshaw, J. M., Acquisti, A., Hoof, R. van, Jeffers, R., & Uszok, A. (2003). Human-Agent Teamwork and Adjustable Autonomy in Practice.

Singer, S. M., & Akin, D. L. (2009). Role Definition and Task Allocation for a Cooperative EVA and Robotic Team, *4970*.

Smith, D. E., Frank, J., & Jonsson, A. K. (2000). Bridging the Gap Between Planning and Scheduling. *The Knowledge Engineering Review*, *15*, 47–83.

Smith, P. J., & Mccoy, C. E. (1999). Alternative Architectures for Distribtued Cooperative Problem-Solving in the National Airspace System.

Sperandio, J. C. (1978). The Regulation of Working Methods as a Function of Work-load among Air Traffic Controllers. *Ergonomics*, *21*, 195–202.

Stanton, N. A., Ashleigh, M. J., Roberts, A. D., & Xu, F. (2001). Testing Hollnagel's Contextual Control Model: Assessing team behaviour in a human supervisory control

task. *International Journal of Cognitive Ergonomics*, *5*.

Stout, R. J., Cannon-Bowers, J. A., Salas, E., & Milanovich, D. M. (1999). Planning, Shared Mental Models, and Coordinated Performance: An Empirical Link Is Established. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *41*, 61–71.

Stubbs, K., Hinds, P. J., & Wettergreen, D. (2007). Autonomy and Common Ground in Human-Robot Interaction: A Field Study. *IEEE Intelligent Systems*, *22*, 42–50.

Varol, Y. L., & Rotem, D. (1981). An Algorithm to Generate All Topological Sorting Arrangements. *The Computer Journal*, *24*, 83–84.

Vicente, K. (1999). *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work*. CRC Press.

Watts, J. C., Woods, D. D., Corban, J. M., Patterson, E. S., Kerr, R. L., & LaDessa, C. H. (1996). Voice loops as cooperative aids in space shuttle mission control. *Computer Supported Cooperative Work '96*, *9*, 48–56.

Woods, D. D. (1988). Coping With Complexity: The Psychology of Human Behavior in Complex Systems. In *Tasks, Errors, & Mental Models* (pp. 128–148).

Woods, D. D. (2002). Essential Characteristics of Resilience. In *Resilience Engineering* (pp. 21–34).

Woods, D. D. (2015). Four concepts for resilience and the implications for the future of resilience engineering. *Reliability Engineering and System Safety*, *141*, 5–9.

Woods, D. D. (2018). Environment Systems and Decisions The theory of graceful extensibility: basic rules that govern adaptive systems. *Environment Systems and Decisions*, *September*.

Woods, D. D., & Hollnagel, E. (2006). *Joint Cognitive Systems: Patterns in Cognitive Systems Engineering* (1st ed.). Boca Raton, FL: CRC Press.

Woods, D. D., Tittle, J., Feil, M., & Roesler, A. (2004). Envisioning Human–Robot Coordination in Future Operations. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *34*.